A Server-Based Signature Solution for Mobile Devices

Thomas Zefferer
Institute for Applied Information Processing and Communications - Graz University of Technology
Inffeldgasse 16a
A-8010 Graz, Austria
thomas.zefferer@iaik.tugraz.at

ABSTRACT

Electronic signatures are frequently used in security-critical fields of application such as e-government or e-banking. During the past years, especially server-based signature solutions have gained popularity, as they solve usability problems of traditional client-based approaches. Unfortunately, server-based signature solutions and their underlying security concepts are usually tailored to classical end-user devices such as desktop PCs or laptops. Therefore, these solutions cannot be used on mobile end-user devices such as smartphones. This excludes an increasing number of potential users, who prefer mobile end-user devices to access e-government and e-banking applications. To solve this problem, we propose a new server-based signature solution that is tailored to the special characteristics of mobile devices. The proposed solution is evaluated by means of a concrete prototype implementation. This prototype proves the feasibility of the proposed solution and demonstrates its capability to leverage the use of electronic signature based applications on mobile end-user devices.

Categories and Subject Descriptors

J.1 [**Computer Applications**]: Administrative Data Processing—*business, financial, government*

General Terms

Security

Keywords

Qualified electronic signatures, server signatures, mobile devices, user authentication

1. INTRODUCTION

Electronic signatures are a key concept of security-critical applications. They rely on asymmetric cryptographic algorithms such as RSA [11] or ECDSA [7] and provide data integrity, authenticity, and non-repudiation or origin. Electronic signatures are especially relevant in countries, in which they have a legal basis. This applies for instance to the European Union (EU). There, the EU Signature Directive [14] defines that so-called qualified electronic signatures (QES) are legally equivalent to handwritten signatures. QES represent a special class of electronic signatures and need to fulfill several requirements. Essentially, QES need to be created in a secure signature creation device (SSCD). SSCDs are typically implemented in secure hardware and need to satisfy security requirements defined in Annex III of the EU Signature Directive [14]. Due to their strong legal basis, QES are frequently used within the EU in various fields of application. For instance, they allow users to carry out transactional e-government procedures or to authorize financial transactions in e-banking solutions. In general, electronic signatures can be useful in any application that requires written consent from users in electronic form.

In most cases, applications do not implement signature-creation functionality on their own. Instead, they rely on separate signature solutions for this purpose. Signature solutions receive signature-creation requests from an application, create the requested signature by interacting with the user, and return the created signature to the application. Current signature solutions can be classified into two categories, depending on the location and implementation of the SSCD. Client-based signature solutions rely on an SSCD that is under physical control of the user. Examples are smart card based solutions such as the Belgian eID card [6] or the Austrian Citizen Card [8]. For client-based signature solutions, each user possesses an own SSCD. In contrast, server-based signature solutions rely on a centrally implemented SSCD that is shared among all users. A prime example is the Austrian Mobile Phone Signature[1], which makes use of a server-based hardware-security module (HSM) that assumes the role of the SSCD. Even though client-based signature solutions have a longer tradition, server-based signature solutions are advantageous in various aspects. Server-based signature solutions define fewer requirements for the mobile end-user device and are hence more feasible. Furthermore, they store and process security-critical data such as cryptographic signing keys in a secure central environment. Malware residing on the local mobile end-user device has hence less opportunities to compromise these data.

Irrespective of the underlying concept, existing signature solutions and their underlying security concepts are tailored

[1] http://www.handy-signatur.at

MoMM'14, December 08 - 10 2014, Kaohsiung, Taiwan
Copyright 2014 ACM 978-1-4503-3008-4/14/12 ...$15.00
http://dx.doi.org/10.1145/2684103.2684142.

to classical end-user devices. During the past years, these devices have however been gradually replaced by mobile end-user devices. Applications—also from security-critical fields of application—react to this trend and progressively adopt this new mobile computing paradigm. Unfortunately, this is difficult for established signature solutions. For instance, smart card based approaches are hardly applicable on mobile devices due to the lack of appropriate card-reading devices. Also server-based signature solutions are difficult to use, as underlying security concepts presume a use of classical end-user devices. So far, no suitable signature solution exists that enables a secure and reliable creation of QES on a mobile end-user device. Hence, applications relying on electronic signatures cannot be used with these devices.

To tackle this problem, we propose a new signature solution, which is tailored to the special characteristics of current mobile end-user devices. As server-based signature solutions are advantageous in various aspects, the proposed solution follows a server-based approach. Basic concepts behind server-based signature solutions are discussed and an abstract model of a server-based signature solution is developed in Section 2. From this model, the secure and reliable authentication of users is identified as most relevant aspect. In Section 3, requirements of user-authentication schemes for server-based signature solutions are derived. Existing authentication solutions for mobile end-user devices are assessed by means of these requirements in Section 4. Based on the results of this assessment, an appropriate authentication scheme for server-based signature solutions is developed in Section 5. By combining this scheme with the defined abstract model of server-based signature solutions, the proposed solution is derived. In Section 6, the applicability and feasibility of this solution is evaluated by means of a concrete implementation. Conclusions are finally drawn in Section 7.

2. SERVER-BASED SIGNATURES

Server-based signature solutions have gained popularity during the past years. In this section, reasons for their growing popularity are discussed and concepts behind server-based signature solutions are identified. From the identified concepts, an abstract model of a server-based signature solution is derived.

2.1 Background

For many years, client-based signature solutions have been the only alternative to create QES according to the EU Signature Directive. These solutions rely on a local hardware-based signature-creation token, which is usually implemented by a smart card. Smart card based signature solutions have been deployed in various European countries such as Austria[2], Belgium[3], or Estonia[4]. While smart card based signature solutions fulfill all functional requirements, they often suffer from low user acceptance due to their poor usability, which is caused by the need for a local card-reading device and software to locally connect and access smart cards. Usability drawbacks of smart card based signature solutions have been discussed by Zefferer et al. [15].

Server-based signature solutions overcome usability drawbacks of client-based approaches. As they rely on a central SSCD, there is no need for special local hardware or software. The remote location of the SSCD is also advantageous in terms of security, as the SSCD can be operated in a secured central environment. There, the SSCD and data stored and processed by the SSCD are immune to malware residing on the user's local device. Due to their various advantages, server-based signature solutions have been a topic of interest for several years. Concrete solutions have early been introduced and discussed for instance by Samadani et al. [12], Bicakci et al. [2] [3], and Ding et al. [4].

Even though server-based approaches are advantageous in various aspects, their suitability for the creation of QES has been in question for several years. Although the EU Signature Directive does not preclude server-based signature solutions a priori, it defines that QES must be *'created using means the signatory can maintain under his sole control'* [14]. As server-based signature solutions store and process the signatory's cryptographic signing key centrally, these means are not under physical control of the signatory. In 2010, Orthacker et al. [9] have shown that sole control over cryptographic signing keys can be guaranteed, even if they are not under physical control of the signatory.

The concept proposed by Orthacker et al. has paved the way for the development of server-based signature solutions that support QES. For instance, the Austrian Mobile Phone Signature implements this concept and has been deployed in Austria on national level. This solution has been in productive operation since 2010 and has proven the applicability of the underlying concept. Additionally, the growing popularity of the Austrian Mobile Phone Signature shows that server-based signature solutions clearly outperform smart card based solutions in terms of user acceptance.

The growing popularity of server-based approaches is also taken into account by legal frameworks that are currently developed in the EU. The EU eIDAS Regulation [5], which will replace the EU Signature Directive, clearly considers the possibility of server-based solutions for the creation of QES. This emancipates server-based signature solutions and assures that their advantages can be fully employed in future.

2.2 Abstract Server-Based Signature Solution

Signature solutions need to satisfy requirements defined by relevant legal frameworks. In Europe, the EU Signature Directive (and in the near future the EU eIDAS Regulation) specifies requirements for the creation of QES. Interestingly, these requirements are quite vague. The Directive mainly specifies the necessity of an SSCD and states that QES must be *'created using means the signatory can maintain under his sole control'* [14]. Regarding the SSCD, the Directive demands in Annex III that *'the signature-creation data used for signature-generation can be reliably protected by the legitimate signatory against the use of others'* and that the SSCD *'must not alter the data to be signed or prevent such data from being presented to the signatory prior to the signature process'* [14]. However, no concrete technical means regarding the fulfillment of these requirements are specified.

Even though requirements defined by relevant legal frameworks are rather vague, a few mandatory properties of server-based signature solutions can still be derived and identified.

- The Creation of QES must be implemented in a remotely implemented SSCD that satisfies the requirements defined in Annex III of the EU Signature Directive [14].

[2]http://www.buergerkarte.at/
[3]http://eid.belgium.be/en/
[4]http://www.id.ee/?lang=en

- Two-factor authentication must be implemented to assure the signatory's sole control over means to create signatures and to reliably protect the signatory's signature-creation data, i.e. cryptographic signing keys, against the use of others.
- The signatory must have the opportunity to review data she is about to sign.

From these properties, an abstract model of server-based signature solutions for the creation of QES can be derived. This model is shown in Figure 1. The model identifies relevant components of the signature solution, interfaces between these components, and external entities interacting with them. External entities are no integral part of the signature solution and are black colored. All components and entities are assigned to one of three domains. Components of the User Domain are implemented locally by the end-user device. As the Service Provider Domain and the Signature-Service Provider Domain are remote, their components are implemented centrally.

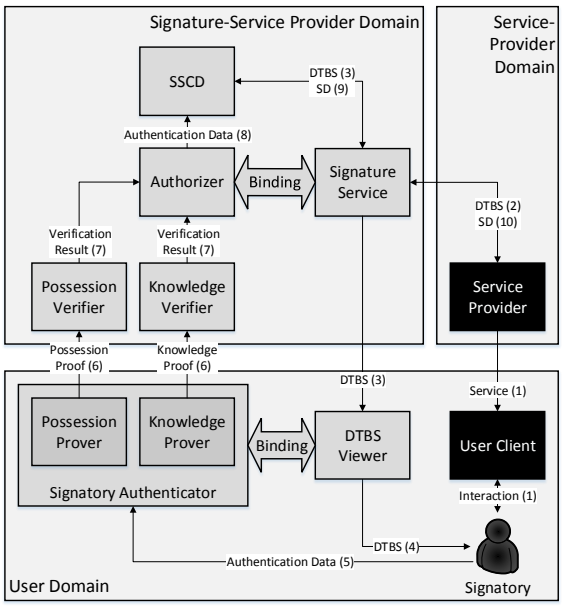

Figure 1: Abstract model for server-based signature solutions.

According to this abstract model, a signature-creation process consists of the following steps. First, the user, i.e. the Signatory, interacts with the User Client to access a service provided by the Service Provider (1). During service provision, the Service Provider requires the Signatory to create a QES. Therefore, the Service Provider contacts the Signature Service to request the signature creation and to transmit the data to be signed (DTBS) (2). The Signature Service forwards the DTBS to the SSCD and displays the DTBS in the local DTBS Viewer (3). This gives the Signatory the opportunity to review the DTBS (4). If the Signatory agrees to sign the displayed DTBS, she authenticates at the signature solution to authorize the signature creation in the remote SSCD. For this purpose, she provides required authentication data to the local Signatory Authenticator (5). From the provided authentication data, the Signatory Authenticator creates a knowledge proof and a possession proof to meet the requirement for a two-factor authentication (6). The knowledge proof and the possession proof are verified in the Signature-Service Provider Domain. Verification results are combined by the Authorizer (7). If all results are positive, signature creation is authorized by supplying the SSCD with required authentication data (8). The signed data (SD) is returned to the Signature Service (9) and forwarded to the Service Provider (10).

There must be a verifiable binding between the displayed DTBS and the authentication data provided by the Signatory Authenticator in the form of knowledge proofs and possession proofs. This binding is mandatory to enable the Signatory to verify that provided authentication data authorize signature creation on the displayed DTBS only. A similar binding is also required between the components Authorizer and Signature Service. This binding enables the signature solution to verify that received authentication data are intended for the current signature-creation process.

2.3 Limitations of Existing Solutions

Relevant legal frameworks define requirements for signature solutions on a rather abstract level. Thus, also the model shown in Figure 1, which has been derived from these requirements, is rather abstract. The only concrete design decision that has been incorporated into this model is the choice of suitable authentication factors. Concretely, the factors knowledge and possession have been chosen to implement the required two-factor authentication. This is reasonable, as the third possible authentication factor inherence, which is usually covered by biometric methods, is known to have several conceptual drawbacks as discussed by Bhattacharyya et al. [1].

Due to its abstract nature, the model shown in Figure 1 is valid for most existing server-based signature solutions. In particular, the model applies to the concept proposed by Orthacker et al. [9] and to the Austrian Mobile Phone Signature. Considering concrete implementations of the derived model, the secure and reliable realization of possession proofs is the challenging part. As the SSCD is implemented remotely, it cannot cover the authentication factor possession. Hence, alternative means need to be implemented to provide required possession proofs. Orthacker et al. propose the use of transaction numbers (TANs) delivered by SMS for this purpose. Following the SMS-TAN approach, the factor possession is covered by the subscriber identity module (SIM) of the Signatory's mobile phone. To verify possession of the SIM, a one-time password, a so-called TAN, is sent to the Signatory's mobile phone. By proving reception of the TAN, the Signatory proves possession of the SIM. The TAN hence represents the required possession proof.

The security of the SMS-TAN approach bases on the assumption that the Signatory uses two separate end-user devices. While a desktop PC or laptop is used to access the Service Provider and to provide required authentication data, the mobile phone is solely used to receive TANs. This doubles the number of devices and communication channels that need to be compromised for a successful attack. Assuming two separate end-user devices, the SMS-TAN approach is hence sufficiently secure. However, this assumption does not comply with use cases involving modern mobile end-user devices. If these devices are used to consume services,

the TAN is received by the same device that is also used to access the Service Provider and to provide authentication data. Therefore, separation of end-user devices and communication channels is not achieved any longer. Thus, the SMS-TAN approach is not suitable for use cases involving modern mobile end-user devices. To overcome this problem, alternative means to cover the authentication factor possession are needed for server-based signature solutions. Requirements for these means are derived in the next section.

3. REQUIREMENTS

Authentication schemes for server-based signature solutions need to fulfill several requirements. These requirements are defined in this section and will be used to systematically assess different user-authentication approaches. All requirements are derived from the abstract model for server-based signature solutions defined in Section 2. According to this model, the two authentication factors knowledge and possession need to be covered. The factor knowledge can be easily implemented by means of passwords. Covering the factor possession is more challenging due to the remote location of the SSCD. For client-based signature solutions, the factor possession is covered by the SSCD itself: the Signatory needs to possess the SSCD to create a signature. For server-based signature solutions, this is not possible, as the Signatory does not possess the SSCD. Hence, other means must be implemented to cover the factor possession.

To systematically identify requirements for implementations of the factor possession, a simplified model is derived from the abstract model shown in Figure 1. This simplified model focuses on components that are directly involved in implementing the authentication factor possession. Furthermore, it considers the assumption that the Signatory uses only one single mobile end-user device, i.e. that all required local components and functionalities are implemented on this device. Accordingly, the User Domain from the underlying general model has been renamed to Mobile Device. The resulting simplified model is shown in Figure 2.

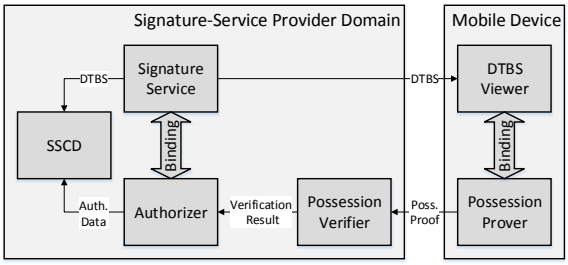

Figure 2: Abstract model of possession-proof implementations.

From this simplified model, a set of general requirements for possession-proof implementations can be derived. Concretely, the following requirements must be met by server-based signature solutions when covering the authentication factor possession.

- **R1: Applicability on mobile end-user device:** All required local components must be feasible and applicable on current mobile end-user devices. Possession-proof implementations must not raise the need for additional local components.
- **R2: Verifiable transaction binding:** There must be a verifiable binding between created possession proofs and the current transaction, i.e. the current signature-creation process. Concretely, both the Signatory and the remote Signature Service must have the opportunity to check whether provided possession proofs are unambiguously linked to the current DTBS.
- **R3: Secure transmission of DTBS:** The confidentiality and integrity of DTBS must be guaranteed during transmission from the remote Signature Service to the local DTBS Viewer. It must be assured that DTBS are not modified before being displayed to the Signatory.
- **R4: Secure transmission of possession proofs:** Possession proofs must be protected during transmission between the local mobile end-user device and the remote Signature-Service Provider Domain. Attackers must not be able to intercept or compromise transmitted possession proofs.

Based on these four requirements, possible approaches to implement the authentication factor possession for server-based signature solutions can be evaluated systematically. Existing approaches to implement possession proofs on mobile end-user devices are surveyed, classified, and assessed in the next section.

4. EXISTING APPROACHES

Implementation of the authentication factor possession using mobile end-user devices has been a topic of interest for several years. Various solutions have been introduced that enable the provision of possession proofs. In this section, these solutions are surveyed and assessed by means of the requirements defined in Section 3. This way, the most suitable approach is determined for this particular use case.

4.1 Survey

During the past years, numerous concepts and solutions to implement the authentication factor possession on mobile end-user devices have been introduced. Many of them resemble each other and follow similar approaches. Hence, most solutions can be classified into a few general categories. These categories are discussed in the following subsections and existing solutions are exemplified for each category.

4.1.1 Static Possession Proofs

Static possession proofs are a simple approach to cover the authentication factor possession. Following this approach, a personalized smartphone app covers the factor possession. This app can be used to send static data, e.g. an ID that is unique for this particular app instance and hence also for the given device. This way, the static data represents a simple possession proof.

Although static possession proofs work in theory, they suffer from several drawbacks. Due to their static nature, the same possession proof is used for each authentication run, which enables replay attacks [13]. Intercepted possession

proofs can be reused for subsequent authentications. Furthermore, static possession proofs cannot be unambiguously linked to a certain authentication run. In summary, static possession proofs are easy to implement but suffer from several conceptual drawbacks. It is hence unsurprising that this approach is hardly followed by security-critical applications to cover the authentication factor possession.

4.1.2 One-Time Passwords

One of the main drawbacks of static possession proofs is their vulnerability to replay attacks. One-time passwords (OTPs) are a common approach to counter this kind of attacks. For each authentication run, a unique and unpredictable OTP is used. As this OTP is valid for one authentication run only, intercepted OTPs cannot be reused.

Implementations of OTP-based approaches often rely on special hardware tokens, which are issued to users and are personalized with a secret cryptographic key. Personalized tokens are paired with the remote entity, at which the user needs to authenticate. This pairing assures that both the remote entity and the user's hardware token share the same cryptographic key. To create OTPs, the token uses the key and dynamic data such as the current time or an internal counter. Only the personalized token is able to access the cryptographic key and hence to create correct OTPs. Thus, proving knowledge of the created OTP also proves possession of the hardware token. The remote entity can verify provided OTPs, as it is paired with the token and hence aware of the required cryptographic key. Productive solutions following the OTP approach are SecurID[5] or DIGIPASS[6]. They all incorporate the current time for the creation of OTPs. A standard to use the current time for the generation of OTPs has been proposed in RFC 6238[7]. Alternatively, also a counter synchronized between the remote entity and the hardware token can be used. This has been described in RFC 4226[8].

The main drawback of the above-mentioned solutions is the need for an additional local hardware token. To render such tokens unnecessary, OTP-based authentication solutions have recently been developed that rely on personalized mobile apps instead of hardware tokens. Examples of such app-based approaches are Google Authenticator[9] or a solution developed by the Barada project[10]. In general, OTP-based approaches are advantageous compared to static possession proofs, as they prevent replay attacks. Still, OTP-based possession proofs are not unambiguously linked to a specific authentication run.

4.1.3 SMS-TAN Approach

The SMS-TAN approach has already been mentioned in Section 2 to illustrate challenges that potentially arise with a use of mobile end-user devices. For the sake of completeness, the SMS-TAN approach is recalled in this section and assessed according to the four identified requirements.

Following the SMS-TAN approach, the authentication factor possession is covered by the user's SIM. Possession of the SIM is verified by sending an OTP—in this concrete use case denoted as TAN—to the user's mobile phone via SMS. By proving reception of the TAN, the user proves possession of the SIM. The SMS-TAN approach is frequently implemented by e-banking solutions to authorize financial transactions. Another popular example that makes use of the SMS-TAN approach is the server-based signature solution Austrian Mobile Phone Signature.

Even though the SMS-TAN approach also relies on OTPs, there is an interesting conceptual difference to OTP-based solutions. Classical OTP-based solutions require only one communication step: the locally generated OTP needs to be transferred to the remote entity. In contrast, the SMS-TAN approach requires two consecutive communication steps. The centrally generated TAN is first transmitted from the remote entity to the user's mobile end-user device. Afterwards, the TAN needs to be transmitted back to the remote entity. While a second communication step might cause additional implementation effort, the central generation of the TAN represents an important advantage. It enables the remote entity to unambiguously bind the TAN to a specific authentication run. By implementing appropriate means, also the user can be enabled to verify this binding. Current implementations of the SMS-TAN approach make use of an additional reference value for this purpose. For instance, the Austrian Mobile Phone Signature displays a unique reference value together with the DTBS. The same reference value is also sent to the user together with the TAN via SMS. This way, the user can verify the binding between the TAN and the DTBS. Thus, in contrast to all other approaches surveyed so far, the SMS-TAN approach establishes a verifiable binding between the current authentication run and provided possession proofs.

Unfortunately, the central generation of the TAN also bears a considerable drawback. The SMS-TAN approach demands that the possession proof, i.e. the TAN, is transferred to the user's mobile end-user device via SMS. However, SMS technology cannot guarantee completely secure data transmissions. This especially applies to smartphones, on which incoming SMS messages can be compromised by malware. Due to its reliance on the potentially insecure SMS technology, the SMS-TAN approach is not able to reliably protect exchanged possession proofs.

4.1.4 Challenge-Response Approaches

Challenge-response approaches are conceptually similar to the SMS-TAN approach, as they also rely on two communication steps. In the first step, the remote entity generates a random challenge and transmits it to the user's mobile end-user device. The mobile device creates a response from this challenge by applying a cryptographic method. This method makes use of a secret cryptographic key, which is specific for a certain mobile device. Thus, responses created from received challenges with this key represent possession proofs. Created responses, i.e. possession proofs, are transmitted to the remote entity in the second communication step. The remote entity cryptographically verifies the obtained response. For this purpose, it must be aware of the cryptographic key that has been used to create the response. Hence, challenge-response approaches require a pairing process, in which relevant key material is exchanged.

Despite existing conceptual similarities of challenge-response approaches and the SMS-TAN approach, two relevant difference can be identified. First, the security of data transmit-

[5] http://www.emc.com/security/rsa-securid.htm
[6] https://www.vasco.com/products/products.aspx
[7] http://tools.ietf.org/html/rfc6238
[8] http://tools.ietf.org/html/rfc4226
[9] https://code.google.com/p/google-authenticator/
[10] http://barada.sourceforge.net/

ted in the first communication step is not relevant for the challenge-response approach. In contrast to the SMS-TAN approach, this data is not a possession proof but merely a required input for the computation of a possession proof. Computation of a valid possession proof is only possible, if the required cryptographic key is known. Intercepting a challenge transmitted during the first communication step does hence not pose a severe threat. This is in stark contrast to the SMS-TAN approach, where TANs transmitted during the first communication step can already be used as valid possession proofs. Second, challenge-response approaches must not necessarily rely on SMS technology. As the authentication factor possession is not covered by the SIM, possession of the SIM does not need to be verified. Hence, the two required communication steps can rely on sufficiently secure communication technologies.

Due to the requirement to locally implement cryptographic functionality, challenge-response approaches were hardly applicable on classical mobile phones. This situation has changed with the emergence of smartphones and other powerful mobile end-user device. During the past years, several authentication solutions following challenge-response approaches have been introduced for mobile end-user devices. These solutions can be classified into two categories. Software-based solutions store required cryptographic key material and implement cryptographic functionality in software. A pure software-based authentication solutions following a challenge-response approach is for instance SQRL[11]. In contrast, hardware-based solutions implement cryptographic functionality and store required keys in secure hardware elements. This provides a higher level of security, but requires mobile end-user devices to provide appropriate hardware components. A concrete example for a hardware-based authentication solution following the challenge-response approach is U2F proposed by the FIDO Alliance[12].

In summary, challenge-response solutions combine the advantages of all other surveyed approaches. Due to the remote generation of challenges, they establish a verifiable binding between the current authentication run and provided possession proofs. Furthermore, they can tolerate the use of potentially insecure communication technologies such as SMS. A comparison of all surveyed approaches by means of the four identified requirements of server-based signature solutions is provided in the following subsection.

4.2 Assessment

The survey of solutions to implement possession proofs on mobile devices has shown that most approaches suffer from conceptual drawbacks. This becomes apparent, when assessing the surveyed approaches by means of the requirements identified in Section 3. Results of this assessment are illustrated in Figure 3.

Figure 3 shows that all surveyed approaches satisfy Requirement R1, which demands applicability on mobile end-user devices. For the conducted survey, only solutions designed for a use on mobile devices have been considered. It is hence unsurprising that Requirement R1 is met by all surveyed solutions. Requirement R2 is only met by the SMS-TAN approach and by challenge-response approaches, as only these solutions establish a verifiable binding between the current authentication run and provided possession proofs. All approaches meet Requirement R3, i.e. provide a secure transmission of DTBS. Suitable technologies to implement a secure communication path between remote entities and local components exist. These technologies can be used to implement a secure transmission of DTBS from the remote Signature Service to the local DTBS Viewer. The situation is slightly different for Requirement R4, which demands a secure transmission of possession proofs. As the SMS-TAN approach requires reliance on the potentially vulnerable SMS technology, a secure transmission of possession proofs is not guaranteed. All other surveyed approaches fulfill this requirement, as they can use secure technologies for the transmission of possession proofs.

	R1	*R2*	*R3*	*R4*
Static Possession Proofs	OK	NOK	OK	OK
One-Time Passwords	OK	NOK	OK	OK
SMS-TAN Approach	OK	OK	OK	NOK
Challenge-Response Approaches	OK	OK	OK	OK

Figure 3: Assessment results.

In summary, obtained assessment results clearly show that challenge-response approaches are the best available choice. They are the only alternative that satisfies all identified requirements. In the next section, we use this finding to develop an improved server-based signature solution that is also applicable on mobile end-user devices.

5. PROPOSED SOLUTION

Challenge-response approaches have been identified to be the best option for the implementation of possession proofs. In this section, a server-based signature solution is proposed that incorporates this finding and that is tailored to the special characteristics of mobile end-user devices. Development of the proposed solution comprises two steps. First, the abstract model of possession-proof implementations shown in Figure 2 is refined. Second, the resulting refined model is integrated into the general abstract model for server-based signature solutions defined in Section 2. This yields a detailed model of the proposed solution.

5.1 Refined Possession-Proof Model

The abstract model of possession-proof implementations derived in Section 3 shows the implementation of possession proofs on a rather abstract level. According to this model, the local Possession Prover and the remote Possession Verifier basically cover the provision of possession proofs. Based on the decision to rely on challenge-response approaches in order to implement the authentication factor possession, these two components can be further detailed yielding the refined model shown in Figure 4.

Essentially, the refined model combines the well-known challenge-response approach with the special requirements and characteristics of server-based signature solutions. According to the refined model shown in Figure 4, the following steps are required to cover the factor possession during a signature-creation process.

I. The Signature Service sends the DTBS to the SSCD

[11] http://sqrl.pl
[12] https://fidoalliance.org/

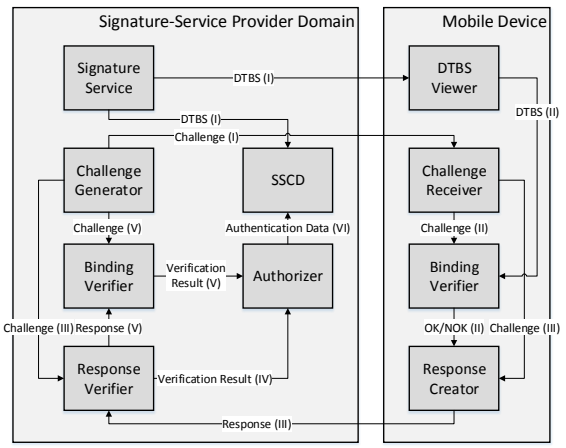

Figure 4: Refined model of possession-proof implementation.

and to the DTBS Viewer. At the same time, a challenge is created by the Challenge Generator, linked to the current signature-creation process, and sent to the Challenge Receiver.

II. The local Binding Verifier checks the binding between displayed DTBS and the received challenge. For this purpose, it fetches required data from the DTBS Viewer and from the Challenge Receiver. The local Binding Verifier forwards the result of the performed check to the Response Creator.

III. If the binding can be positively verified, the Response Creator creates a response from the received challenge. For this purpose, it fetches the challenge from the Challenge Receiver. Creation of the response is based on a secret cryptographic key, which covers the authentication factor possession. The created response is transmitted to the remote Response Verifier.

IV. The Response Verifier verifies the possession proof. The verification result is forwarded to the Authorizer.

V. The remote Binding Verifier checks, whether the received possession proof is linked to the current signature-creation process. For this purpose, it fetches the challenge from the Challenge Generator and the response from the Response Verifier. The result of the performed check is forwarded to the Authorizer.

VI. The Authorizer combines all received verification results. If they are all positive, it authorizes the signature-creation process in the SSCD by providing the required authentication data.

5.2 Enhanced Signature Solution

The refined model of possession-proof implementations can be integrated into the general abstract model of server-based signature solutions defined in Section 2. This way, an enhanced signature solution can be derived, which is ready to be used with mobile end-user devices. This enhanced solution is shown in Figure 5.

Even though the proposed solution resembles the general abstract model for server-based signature solutions defined in Section 2, some relevant differences can be identified. First, components related to the provision of possession proofs have been replaced according to the refined model of possession-proof implementations. Second, the User Domain from the abstract model has been replaced. In the proposed solution, all local components are implemented on one mobile device, which is used by the Signatory. Third, the proposed solution defines the Service Provider to be implemented on the mobile device as well. This is reasonable, as provision of functionality by means of local mobile apps is common practice on mobile end-user devices. Such mobile apps combine the functionalities of the Service Provider and the User Client. Nevertheless, the proposed solution is also applicable to use cases with remote Service Providers such as web applications. For the sake of simplicity, only the more likely case of a local Service Provider is considered here in detail.

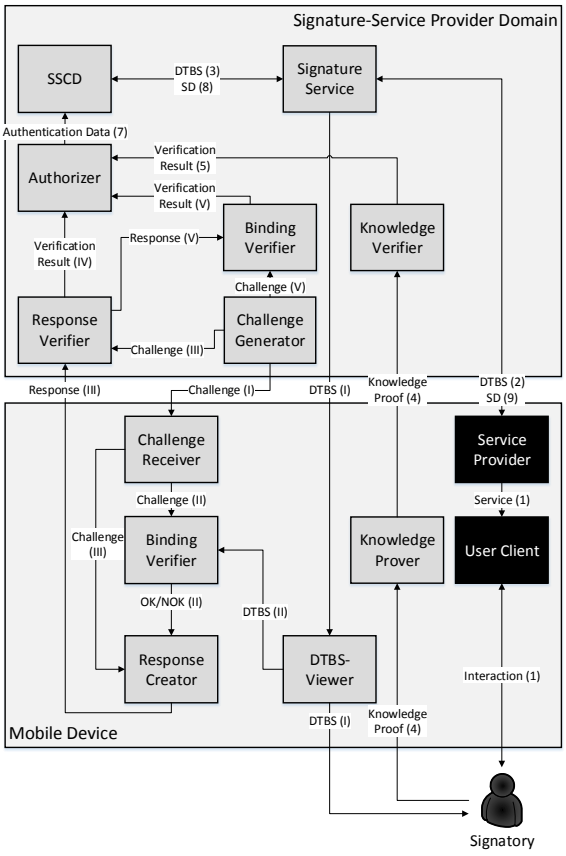

Figure 5: Enhanced signature solution for mobile device.

According to the proposed solution shown in Figure 5, a typical signature-creation process comprises the following steps.

1. By interacting with the User Client, the Signatory consumes a service provided by the Service Provider. Considering the illustrated case of a local Service Provider, the Signatory uses a mobile app running on his or her

mobile end-user device.

2. The Service Provider, i.e. the mobile app, requires the Signatory to create an electronic signatures. Therefore, it sends a signature-creation request with the DTBS to the Signature Service.
3. The Signature Service provides the SSCD with the received DTBS.
4. To authenticate the Signatory, a knowledge proof is requested from the Signatory first. The Signatory provides this knowledge proof to the Knowledge Prover, which forwards it to the Knowledge Verifier.
5. The Knowledge Verifier verifies the knowledge proof. The verification result is forwarded to the Authorizer.
6. A possession proof is requested from the Signatory. To provide the possession proof, the processing steps I-V as described in Section 5.1 are carried out.
7. The Authorizer combines all received verification results. If all results are positive, the signature-creation process is authorized in the SSCD.
8. The SSCD creates an electronic signature on the DTBS and returns the signed data (SD) to the Signature Service.
9. The Signature Service creates a signature-creation response containing the signed data and sends this response to the Service Provider.

In contrast to existing server-based signature solutions, the proposed solution can also be used on mobile end-user devices. Due to reliance on a challenge response based authentication mechanism, it overcomes limitations of solutions using SMS-TAN approaches. The applicability and feasibility of the proposed solution has been evaluated by means of a prototype implementation, which is introduced in the next section.

6. EVALUATION

The proposed server-based signature solution for mobile end-user devices defines relevant components and specifies reliance on a challenge-response based authentication scheme. However, the proposed solution is still rather abstract and does not define concrete realizations of identified components and communication paths. In this section, we introduce a prototype implementation of the proposed solution, in order to evaluate its feasibility and applicability.

6.1 Design Decisions

Development of the prototype implementation has been based on a set of design decisions, which specify the concrete realization of relevant components. In particular, the following design decisions have been made.

- The authentication factor knowledge is covered by alphanumeric passwords, in order to rely on a familiar and approved method.
- Challenges are implemented by TANs sent to the mobile device via SMS. As the confidentiality of challenges is irrelevant for the security of provided possession proofs, reliance on SMS technology for the delivery of challenges is acceptable.
- The local binding between DTBS and possession proofs is verified manually by the Signatory with the help of reference values. This gives the Signatory more control over the entire signature-creation process.
- The creation of responses from challenges is based on asymmetric cryptographic methods. Concretely, received TANs are cryptographically signed on the mobile device with a cryptographic key. Accordingly, created signed TANs represent possession proofs. Although this design decision imposes an additional local signature creation, the local signing of the TAN does not carry the concept of server-based signatures ad absurdum. The relevant QES is still created in a secure server-based environment. In contrast, the local signature of the TAN does not need to meet the strict requirements of QES and can be implemented using the best suitable technologies available on the particular mobile end-user device.

By applying these design decisions to the proposed solution shown in Figure 5, a functional model of the implemented prototype can be derived. This functional model is discussed in the following subsection.

6.2 Functional Model

The functional model of the developed prototype implementation is shown in Figure 6. It resembles the proposed abstract solution shown in Figure 5, but further details several components according to the made design decisions.

Assuming a local Service Provider, the functional model combines the components User Client and Service Provider in one single component Service Provider App. The Service Provider App provides a service to the Signatory, defines the DTBS, triggers the signature creation, and receives SD. Considering the design decision to cover the authentication factor knowledge by means of alphanumeric passwords, the functional model replaces the abstract components Knowledge Prover and Knowledge Verifier with the concrete components Password Requester and Password Verifier.

Based on the decisions to realize challenges by means of SMS TANs, to create responses by signing these TANs, and to leave the verification of required bindings between DTBS and possession proofs to the Signatory, components involved in the provision of possession proofs can be further refined. The functional model replaces the abstract Challenge Generator with the concrete component TAN Generator, which generates a random TAN and a reference value. Both are sent to the mobile device via SMS. There, the SMS Inbox App of the mobile device receives the SMS and hence implements the functionality of the abstract Challenge Receiver.

With the DTBS Provider, the functional model introduces a new component in the Signature-Service Provider Domain. This component is necessary for the integration of reference values, which enable a manual transaction-binding verification. The DTBS Provider combines the DTBS obtained from the Signature Service with the reference value created by the TAN Generator and sends both to the DTBS Viewer. The Signatory compares the reference value displayed with the DTBS with the one received together with the TAN to verify the binding between the DTBS and the received TAN. Afterwards, the Signatory enters the received TAN to the TAN Signer. The TAN Signer implements the functionality of the abstract Response Creator and signs the provided

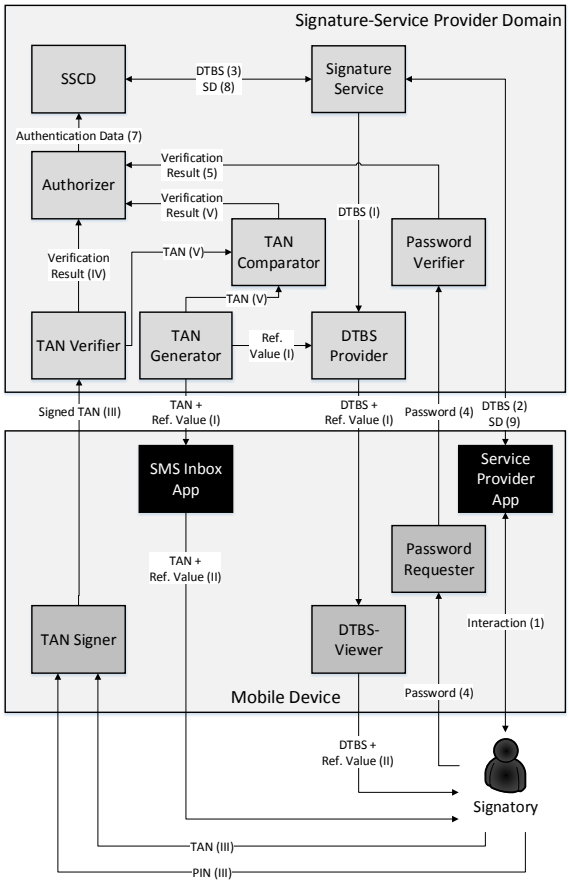

Figure 6: Functional model of the prototype implementation.

TAN with the private key of a Signatory-specific key pair. Access to this key is protected by a PIN, which needs to be entered by the Signatory.

In the Signature-Service Provider Domain, the TAN Verifier implements the functionality of the abstract Response Verifier. The TAN Verifier verifies the received signed TAN using the corresponding public key. The received TAN is also forwarded to the TAN Comparator, which assumes the role of the abstract remote Binding Verifier. The TAN Comparator checks if the received TAN corresponds to the generated TAN. Verification results from the Password Verifier, TAN Verifier, and TAN Comparator are finally combined by the Authorizer, which finally authorizes the signature-creation process in the remote SSCD.

6.3 Prototype Implementation

The functional model represents the basic architecture of the developed prototype. Accordingly, the prototype comprises a server component, which implements components of the Signature-Service Provider Domain, and a smartphone app, which covers functionality assigned to the mobile device.

To reduce development time, the prototype has been based on an existing server-based signature solution called ServerBKU. This solution has been introduced by Rath et al. [10] and basically resembles the Austrian Mobile Phone Signature. The ServerBKU supports the server-based creation of electronic signatures and thus already implements the functionality of the components Signature Service and SSCD defined by the functional model. As it relies on the concept proposed by Orthacker et al. [9], the ServerBKU makes use of the classical SMS-TAN approach to authenticate users. The authentication factor possession is covered by a TAN, which is sent to the user via SMS and has to be returned by the user to prove possession of the SIM. As the underlying security concept of this approach assumes the use of two separate end-user devices, the ServerBKU cannot be used on one single mobile devices. The developed prototype extends the ServerBKU by replacing its SMS TAN based user authentication with the challenge response based method defined by the proposed solution. This way, the ServerBKU is prepared to be used on a single mobile end-user device.

To integrate the proposed solution, several modifications have been applied to server components of the ServerBKU. In particular, the TAN Verifier component has been added, which cryptographically verifies the validity of received signed TANs. For this purpose, the ServerBKU's user-registration and signature-creation processes have been extended. During the registration process, the ServerBKU creates an account for the Signatory, assigns a unique ID to this account, generates required keys for the creation of electronic signatures, asks the Signatory to define a secret password for this account, and issues a signing certificate for the Signatory. This process has been extended such that an additional asymmetric key pair used for signing TANs is generated on the mobile device. The Signatory is also asked to define a PIN that protects this key pair. The public part of this key pair is stored together with other Signatory-related data in the created user account. During the signature-creation process, the ServerBKU authenticates the Signatory before creating an electronic signature. This user authentication has been extended such that the ServerBKU expects reception of a signed TAN and is able to cryptographically verify it with the help of the public key stored during the registration process.

In addition to extending server components of the ServerBKU, also required functionality assigned to the mobile device has been implemented. All required functionality is covered by a mobile Google Android app called SignedTAN App. Google Android[13] has been chosen as preferred platform for the prototype due to its popularity and wide spread[14]. According to the functional model shown in Figure 6, this app needs to cover the functionality of the Password Requester, the DTBS Viewer, and the TAN Signer. SMS-receiving functionality is implemented by the SMS Inbox App that comes with Google Android and does not need to be covered by the SignedTAN App. Similarly, also the Service Provider App, which basically covers the functionality of the Service Provider, is not part of the SignedTAN App.

During the ServerBKU's registration process, the SignedTAN App generates the key pair that is later used to sign TANs, stores the private part of the key pair, asks the Signatory to define a secret PIN to protect the private key, and sends the public part of the key to the ServerBKU. This way, the SignedTAN App is paired with the user's ServerBKU account. During signature-creation processes, the SignedTAN

[13]http://www.android.com/

[14]http://mobithinking.com/mobile-marketing-tools/latest-mobile-stats/a

App provides user interfaces (UIs), through which required data can be entered by the user. Screenshots of relevant UIs provided by the SignedTAN App are illustrated in Figure 7.

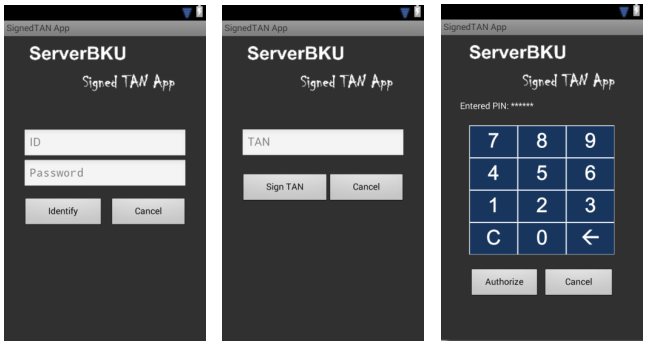

Figure 7: User interfaces of the SignedTAN App.

The leftmost screenshot shows the UI through which users can start the authentication process by entering ID and password. After positive verification of the entered password, the user receives a SMS with a TAN. This TAN can be entered to the SignedTAN App using the UI shown in the middle screenshot. The entered TAN is afterwards signed by the SignedTAN App. To authorize access to the locally stored key that is required to sign the TAN, a PIN needs to be entered by the user. For this purpose, the SignedTAN App provides the UI shown in the rightmost screenshot.

The prototype implementation is fully functional and proves that the proposed server-based signature solution can be realized with available technologies. It also shows that the improved authentication and authorization scheme does not raise severe efficiency and usability issues. In particular, the required cryptographic operations do not cause significant delays. This way, the proposed solution and the developed prototype pave the way for a secure and convenient usage of server-based signature solutions on mobile end-user devices.

7. CONCLUSIONS

In this paper, we have proposed a new server-based signature solution. While concepts of server-based signature solutions have been known for several years, existing solutions have mainly been designed for classical end-user devices. Hence, they are often not applicable on mobile end-user devices. The proposed signature solution solves this problem by integrating a challenge response based authentication scheme that is tailored to the special characteristics of modern mobile end-user devices. The applicability and feasibility of the proposed solution has been evaluated and demonstrated by means of a prototype implementation.

The developed prototype shows that the proposed solution can be easily integrated into existing server-based signature solutions that have originally been designed and developed for classical end-user devices. Thus, the proposed solution offers a convenient opportunity to make these solutions ready for a use on mobile end-user devices. This finally leverages the mobile use of electronic signature based applications from security-critical fields such as e-government or e-banking, and assures that these applications are no longer restricted to classical end-user devices.

8. REFERENCES

[1] D. Bhattacharyya, R. Ranjan, A. Farkhod Alisherov, and M. Choi. Biometric authentication: A review. *International Journal of u-and e-Service, Science and Technology*, 2(3):13–28, 2009.

[2] K. Bicakci and N. Baykal. {SAOTS:} A New Efficient Server Assisted Signature Scheme for Pervasive Computing. In *Security in Pervasive Computing*, pages 187–200. 2004.

[3] K. Bicakci and N. Baykal. Improved server assisted signatures. *Computer Networks*, 47(3):351–366, 2005.

[4] X. Ding, D. Mazzocchi, and G. Tsudik. Experimenting with Server-Aided Signatures. In *Proceedings of the Symposium on Network and Distributed Systems Security (NDSS 2002)*. Internet Society, 2002.

[5] European Commission. Proposal for a REGULATION OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL on electronic identification and trust services for electronic transactions in the internal market, 2012.

[6] A. Fairchild and B. de Vuyst. The Evolution of the e-ID card in Belgium: Data Privacy and Multi-Application Usage. In *6th Int. Conference on Digital Society*, pages 13–16, Valencia, 2012.

[7] ISO/IEC. ISO/IEC 14888-3:2006 – Information technology – Security techniques – Digital signatures with appendix – Part 3: Discrete logarithm based mechanisms, 2006.

[8] H. Leitold, A. Hollosi, and R. Posch. Security architecture of the Austrian citizen card concept. In *Computer Security Applications Conference, 2002. Proceedings. 18th Annual*, pages 391–400, 2002.

[9] C. Orthacker, M. Centner, and C. Kittl. Qualified Mobile Server Signature. In *Proceedings of the 25th TC 11 International Information Security Conference SEC 2010*, 2010.

[10] C. Rath, S. Roth, M. Schallar, and T. Zefferer. A Secure and Flexible Server-Based Mobile eID and e-Signature Solution. In *The Eighth International Conference on Digital Society*, pages 7–12, 2014.

[11] R. L. Rivest, A. Shamir, and L. Adleman. A Method for Obtaining Digital Signatures and Public-key Cryptosystems. *Commun. ACM*, 21(2):120–126, 1978.

[12] M. H. Samadani, M. Shajari, and M. M. Ahaniha. Self-proxy mobile signature: A new client-based mobile signature model. In *24th IEEE International Conference on Advanced Information Networking and Applications Workshops, WAINA 2010*, pages 437–442, 2010.

[13] P. Syverson. A Taxonomy of Replay Attacks. In *In Proceedings of the 7th IEEE Computer Security Foundations Workshop*, pages 187–191. Society Press, 1994.

[14] The European Parliament and the Council of the European Union. DIRECTIVE 1999/93/EC OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 13 December 1999 on a Community framework for electronic signatures, 1999.

[15] T. Zefferer and V. Krnjic. Usability Evaluation of Electronic Signature Based E-Government Solutions. In *Proceedings of the IADIS International Conference WWW/INTERNET 2012*, pages 227–234, 2012.