Design and Application of a Secure and Flexible Server-Based Mobile eID and e-Signature Solution

Christof Rath, Simon Roth, Manuel Schallar, and Thomas Zefferer
Institute for Applied Information Processing and Communications
Graz University of Technology
Graz, Austria
Email: {first name}.{last name}@iaik.tugraz.at

***Abstract*—Electronic identities (eID) and electronic signatures are basic concepts of various applications and services from security-critical domains including e-government, e-business, and e-commerce. During the past years, server-based approaches have been increasingly followed to implement these concepts. Unfortunately, existing server-based eID and electronic-signature solutions are usually tailored to a specific use case or deployment scenario. This renders a deployment of these solutions in arbitrary application scenarios difficult. To overcome this issue, we propose a flexible server-based eID and electronic-signature solution that can be easily deployed in arbitrary application scenarios while still providing a sufficient level of security and usability. The feasibility of the proposed solution is demonstrated by means of a concrete implementation. Furthermore, the claimed flexibility of the developed solution is shown by integrating it into a productive web-based time-tracking application. Its successful deployment and integration shows that the proposed solution provides a secure and flexible alternative to existing eID and electronic-signature solutions and that it has the potential to improve the security of security-critical services and applications from arbitrary domains.**

Keywords*—*e-government, e-business, eID, electronic identity, electronic signature, identity management, mobile security.

I. INTRODUCTION

With the rise of digital society, remote identification of users has become an increasing challenge as a growing number of services have been moved to the Internet. Design and development of concepts and solutions that provide remote identification of users have been a topic of interest for many years. We have recently contributed to this topic and have proposed and presented a server-based eID and electronic-signature solution that facilitates remote identification of users in arbitrary application scenarios [1]. In this article, we further delve into this topic and elaborate on our proposed solution.

In general, the need for reliable remote identification of users applies to public-sector applications (e-government) as well as to private-sector applications (e-commerce, e-business). Remote identification is usually achieved by means of a unique eID assigned to the user. An eID can for instance be a unique number, user name, or e-mail address. During authentication, the claimed identity (eID) is proven by the user. Reliance on secret passwords for authentication purposes is still the most popular and most frequently used authentication approach for online services. However, password-based authentication schemes have turned out to be insecure due to their vulnerability against phishing attacks and their poor usability, which often leads to the use of weak passwords that are easy to guess or easy to break [2][3].

Transactional online services from the e-government domain and related fields of application typically require reliable remote identification and authentication of users. Given the obvious drawbacks of password-based eID and authentication schemes in terms of security, two-factor authentication schemes have been developed for applications with high security requirements such as transactional e-government services. Current two-factor authentication schemes typically comprise the authentication factors *possession* and *knowledge*.

Popular examples of two-factor authentication schemes are smart card based solutions. During the authentication process, the user proves to be in *possession* of the eID token (i.e., the smart card) and proves *knowledge* of a secret PIN (personal identification number) that is specific to this eID token and that protects access to the token and to eID data stored on it. In most cases, smart cards additionally enable users to create electronic signatures (e-signatures). For this purpose, the smart card additionally stores a secret signing key and features hardware-based signature-creation capabilities. Access to the signing key and to the smart card's signature-creation functionality is again protected by means of two-factor authentication.

Smart cards are an ideal technological choice to combine the concepts of eID and e-signature, as they are capable to implement both eID and e-signature functionality. Thus, they are frequently used in security-critical fields of application such as e-business, e-banking, or e-government. For instance, various transactional e-government services that have been launched in Europe during the past years require users to authenticate themselves remotely with a personalized smart card and to complete online transactions by applying an electronic signature with the same card [4]. Unfortunately, smart card based solutions usually lack an appropriate level of usability, as they require users to obtain, install, and use an appropriate card-reading device in combination with the associated software [5].

Powered by the recent emergence of mobile communication technologies and motivated by the low user acceptance of smart card based eID and e-signature solutions, several *mobile* eID and e-signature solutions have been developed during the past years [6]. These solutions render the use of smart cards unnecessary, as they cover the authentication factor *possession*

by means of the user's mobile phone. This way, mobile eID and e-signature solutions have the potential to significantly improve usability while maintaining a comparable level of security to smart card based solutions. This is supported by the fact, that, e.g., in Austria qualified signatures can be issued both, with smart cards and with mobile eID and e-Signature solutions.

Due to their improved usability compared to smart card based authentication schemes [5], mobile eID and e-signature solutions are in principle also suitable for use cases with lower security requirements. Unfortunately, existing mobile eID and e-signature solutions are usually tailored to the requirements of specific use cases and fields of application. This applies to most mobile eID and e-signature solutions that have been introduced and launched worldwide during the past years. Due to their limitation to specific use cases, these solutions can hardly be used in different fields of application. This leads to situations, in which most applications cannot benefit from the enhanced security and usability of existing mobile eID and e-signature solutions.

To overcome this problem, we propose a modular and flexible concept for mobile eID and e-signature solutions. The main idea behind the design of the proposed concept was to achieve a flexible solution and to maintain its compatibility to different use cases and application scenarios. Details of the proposed concept are presented in this article.

In Section II, we start with a brief survey of existing mobile eID and e-signature solutions and discuss their strengths and limitations. We then derive requirements of a mobile eID and e-signature solution that is applicable in arbitrary application scenarios in Section III. In Section IV, we introduce a technology-agnostic architecture for a mobile eID and e-signature solution that meets all predefined requirements. Based on the proposed architecture, we model three technology-agnostic processes that cover required functionality in Section V. The practical applicability and feasibility of the proposed solution is assessed in Section VI by means of a concrete implementation. The compatibility of this implementation with existing security-critical applications is evaluated in Section VII. Finally, conclusions are drawn in Section VIII.

II. Related Work

The reliable remote identification and authentication of users by means of two-factor based approaches has been a topic of interest for several years. For many years, smart cards have been the preferred technology to implement two-factor based authentication schemes. Thus, smart card based solutions have been introduced in several security-sensitive fields of application during the past decades. Especially in Europe, various countries, such as Austria [7], Estonia [8], Belgium [9], or Spain [10] have issued personalized smart cards to their citizens in order to reliably identify and authenticate them during transactional e-government procedures [4]. In most cases, smart cards do not only provide eID functionality but also enable users to create electronic signatures. This is of special importance in Europe, where electronic signatures can be legally equivalent to handwritten signatures according to the EU Directive 1999/93/EC [11]. The importance of electronic signatures is even strengthened by the EU Regulation on electronic identification and trusted services for electronic transactions in the internal market [12], which will soon replace EU Directive 1999/93/EC.

While smart cards work fine from a functional point of view, their usability is usually rather poor. This poor usability is mainly caused by the need for a card-reading device to physically connect the smart card to the user's computer. The need for additional drivers and software to communicate with the smart card and to integrate its functionality into security-critical applications also decreases the usability of smart-card technology in general and of smart card based eID and e-signature solutions in particular. This has for instance been shown by Zefferer et al. [5], who have set up a thinking-aloud test with 20 test users to determine and compare the usability of different approaches to provide eID and e-signature functionality. The conducted usability test has shown that users clearly prefer solutions that do not require smart cards and card-reading devices.

To overcome usability limitations of smart card based solutions, several mobile two-factor based eID and e-signature solutions have been developed during the past years. Surveys of mobile eID and e-signature solutions have for instance been provided by Ruiz-Martinez et al. [6] and Pisko [13]. All these solutions have in common that the factor *possession* is not covered by a smart card but by the user's mobile phone. All mobile eID and e-signature solutions that comply with demanding legal requirements, such as those defined by the EU Signature Directive, include some kind of secure hardware element, which is able to securely store eID data and to carry out cryptographic operations. Depending on the realization and location of this secure hardware element, mobile eID and e-signature solutions can be basically divided into the following two categories:

1) **SIM-based solutions:** Solutions belonging to this category make use of the mobile phone's SIM (subscriber identity module) to securely store eID data and to carry out cryptographic operations. In most cases, the use of a special SIM is required, as off-the-shelf SIMs do not feature the required cryptographic operations. Access to eID data stored on the SIM and to cryptographic functionality provided by the SIM is typically protected by a secret PIN that is only known to the legitimate user. This way, SIM-based solutions rely on two different authentication factors. This PIN covers the factor *knowledge* of the two-factor based authentication scheme. The factor *possession* is covered by the SIM itself, which is under physical control of the user. With regard to security, all SIM-based solutions share one conceptual drawback. As required cryptographic operations such as the creation of electronic signatures are carried out on the mobile end-user device, these operations and all the data that is processed by these operations are potentially prone to malware residing on this device. This is especially an issue on current popular smartphone platforms such as Android, which are known to be vulnerable against malware [14].
2) **Server-based solutions:** Server-based mobile eID and e-signature solutions implement the secure hardware element centrally, e.g., in a hardware security module (HSM) at the service provider. Such a solution has been proposed by Orthacker et al. [15]. The user's mobile phone does neither implement cryptographic functionality, nor store eID data. However, the mobile phone is an integral component of the

authentication process that is mandatory in order to gain access to centrally stored eID data and to carry out electronic signatures. Server-based solutions rely on two authentication factors. During signature-creation processes, the user needs to provide a secret password first. This password covers the authentication factor *knowledge*. Covering the authentication factor *possession* is more challenging. As the server-based secure hardware element is not under physical control of the user, this element cannot cover the authentication factor *possession*. This factor is again covered by the user's mobile phone, concretely by the user's SIM. To complete the authentication process, a one-time password is sent to the user's mobile device via SMS. This one-time password has to be returned by the user. This way, the user proves *possession* of the SIM, as the one-time password can only be received, if the user has control over the SIM. With regard to security, server-based approaches are conceptually advantageous, as they do not require critical data to present on potentially insecure and compromised mobile end-user devices. The weakest point of the server-based signature solution presented by Orthacker et al. [15] is probably the SMS-based user-authentication step, as SMS messages must not be assumed to be secure on certain smartphone platforms any longer [14].

For above-mentioned categories, concrete mobile eID and e-signature solutions have been developed and rolled-out on a large scale. For instance, SIM-based mobile eID and e-signature solutions have been set into productive operation in Estonia [16] and Norway [17]. A server-based mobile eID and e-signature solution has been in productive operation in Austria since 2009 [18]. Most existing solutions are tailored to a specific legal framework (e.g., national laws) or to a certain identity system (e.g., to a specific national eID system). For instance, the Austrian mobile eID and e-signature solution has been purpose-built for the Austrian official eID infrastructure and bases on data structures, protocols, and registers that are specific to the Austrian use case. The Austrian eID infrastructure has been discussed by Stranacher et al. [19] in more detail. Deploying this purpose-built solution in other countries would require major adaptations and cause additional costs. Similar limitations apply to most mobile eID and e-signature solutions that have been set into productive operation so far. Their purpose-built nature renders a use of these solutions in different fields of application difficult and expensive. This prevents a broad roll-out of mobile eID and e-signature solutions and prevents that all applications can benefit from their improved security and usability.

III. Requirements

The conducted survey on existing mobile eID and e-signature solutions has identified a lack of dynamically adaptable solutions that can easily be applied to arbitrary use cases. To tackle this issue, we propose a mobile eID and e-signature solution that can easily be used in arbitrary application scenarios. We have designed the proposed solution according to a set of requirements. These requirements have been extracted from an analysis of existing solutions and from published evaluations of these solutions such as the one presented in [5]. The derived requirements (R1-R5) are discussed in the following in more detail.

- **R1: Flexibility regarding external components:** Mobile eID and e-signature solutions typically rely on external parties and components. Common examples for such components are certification authorities (CA), which bind a user's identity to her signing key, or identity databases (e.g., official person registers or company databases), which are required to derive eIDs for users. A generic mobile eID and e-signature solution must not be limited to certain external components but provide flexible means to integrate different external components (e.g., different CAs).
- **R2: Avoidance of token roll-outs:** Long-term experience with smart card based solutions has shown that the roll-out of eID and e-signature tokens (e.g., smart cards, SIMs) causes additional (financial) effort and hence reduces user acceptance. Avoidance of necessary roll-outs of such tokens is hence a key requirement for usable mobile eID and e-signature solutions.
- **R3: Usability:** The often disappointing user acceptance of smart card based solutions shows that usability is an important success factor of eID and e-signature solutions. For mobile eID and e-signature solutions, the following aspects need to be considered in particular in order to achieve an appropriate level of usability:
  - **R3a: Avoidance of installations:** Usable solutions must not require the user to obtain, install, and maintain additional hardware or software, as this causes additional effort.
  - **R3b: Platform and device independence:** Usable solutions must not be restricted to certain computing platforms, operating systems, or end-user devices, as users want to access services everywhere and at any time irrespective of their current execution environment.
  - **R3c: Location independence:** Usable mobile eID and e-signature solutions must not be bound to a certain mobile network but must also be accessible when roaming in foreign networks.
- **R4: Security:** Security is an important requirement, as mobile eID and e-signature solutions are mainly applied in security-sensitive fields of application such as e-government or e-commerce. Hence, mobile solutions must assure a comparable level of security to other two-factor based eID and e-signature solutions and must be able to comply with given legal requirements such as the EU Signature Directive [11].
- **R5: Easy and flexible deployment and operation:** From the service operator's point of view, mobile signature solutions should support an easy and flexible deployment as well as an efficient operation, in order to save installation, set-up, and operation costs.

Based on these requirements, we propose a generic and adaptable mobile eID and e-signature solution, which removes limitations of existing solutions. We introduce and discuss the concept of our solution in the next sections before providing details on its implementation in Section VI.

IV. Architecture

Mobile eID and e-signature solutions follow either a SIM-based or a server-based approach to store eID data and to create electronic signatures. Other approaches would be possible on

smartphones but cannot be applied on standard mobile phones due to their limited capabilities. Considering the requirements defined in Section III, we have decided to follow a server-based approach for our solution. This means, that a central HSM is responsible for protecting all eID data as well as for computing electronic signatures. Since solutions based on server-side signatures have very limited hardware requirements on the user side, they are comparatively cheap, user-friendly, and flexible in their deployment, as no roll-out of tokens is required. This way, Requirement R2 and Requirement R5, which demand avoidance of token roll-outs and an easy and flexible deployment and operation, are fulfilled.

Furthermore, server-based approaches require no up-front investments in dedicated SIM cards and no requirements towards the mobile network operators (MNO), hence, the targeted user group is not limited to a single, or certain MNOs. This reduces barriers and enhances usability. Advantages of server-based signature-creation approaches in terms of usability and user acceptance have also been discussed by Zefferer et al. [5]. Thus, reliance on a server-based approach assures that Requirement R3, which demands a sufficient level of usability, is met.

A theoretic concept of a server-based mobile signature solution and an approach to store users private keys in a secure manner on a remote server have been proposed by Orthacker et al. [15]. The proposed solution fulfills the requirements of *qualified electronic signatures* as defined by EU Directive 1999/93/EC [11], which emphasizes the suitability of this concept for security-critical application scenarios. Furthermore, a server-based mobile eID and e-signature solution that is compliant to the EU Directive 1999/93/EC has been in productive operation in Austria for several years. This provides evidence that server-based solutions are capable to achieve a sufficient level of security and hence to meet Requirement R4.

On a high level view, our solution defines the three processes: *registration*, *activation* and *usage*. These processes have different properties regarding computational effort and security constraints. During registration, which is mainly a matter of legal and organizational requirements, the identity of the user is verified. Usually, it is sufficient to perform the registration only once per user. During activation, a new eID including a signing key and a certificate is created for a registered user. Activation is required once per life span of an eID. In the usage process, created eIDs and signing keys are used by the user for authentication purposes and to create electronic signatures. Details of the three processes will be provided in the following section.

The architecture of our mobile eID and e-signature solution reflects the three processes defined above. This is illustrated in Figure 1. The entire architecture is split into an inner part and an outer part. Components implementing functionality of the activation and the usage processes are executed within these two parts. As shown in Figure 1, each part has its own database to store required internal data.

This way, the architecture is mainly composed of two databases and the four core components *Activation Outer*, *Activation Inner*, *Usage Outer*, and *Usage Inner* as well as a central HSM as inner component. The split between inner and outer components is a security feature as it reduces the impact of a data loss in case a service connected to the outer world gets compromised. Communication between outer and inner components happens via a limited, pre-defined set of commands over an encrypted channel. The separation of the core components allows for a very flexible deployment where, e.g., the activation parts can run on different machines, a different network or, if the business process allows/demands it, without a remote access at all. Additionally, access rights can be granted more restrictively, as only the activation process requires write access to many fields in the databases. At the same time, it is also possible to deploy the complete service on a single machine, if this is the preferred deployment scenario. By defining separate components to cover the proposed solution's functionality, the chosen architecture meets Requirements R4 and Requirement R5, which demand a sufficient level of security as well as an easy and flexible deployment and operation.

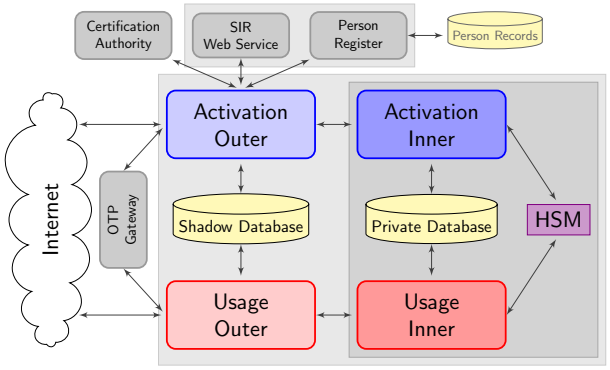


Figure 1: Overview of Core Components.

In addition to the four core components, the two databases and the HSM, the proposed architecture defines two internal and two external components. The external component *OTP Gateway*, which stands for one-time password gateway, is required during the registration, activation and usage process to send OTPs or activation codes to users. The internal component *SIR Web Service* is necessary for receiving so-called *Standard Identification Records* (SIRs). These records enable offline registrations, which will be discussed in detail in the next section. The components *Person Register* and *Certification Authority (CA)* are required during the activation process. While the CA is an external component, the Person Register is an internal component, which usually connects to an external database. The purpose of these components will also be discussed in detail in the next section. By clearly separating these components from the core components of the proposed solution, Requirement R1, which demands flexibility regarding external components, is already fulfilled on architectural level. The three processes, which build up our solution and cover its functionality, as well as all involved components, are described in the following section in detail.

V. Processes

The entire functionality of the proposed technology-agnostic mobile eID and e-signature solution is covered by the processes *registration*, *activation* and *usage*. The purpose of these processes is discussed in the following subsection in more detail.

A. Registration Process

During the registration process, data necessary to unambiguously identify a user is collected. Each user has to

run the registration process at least once, before being able to use the proposed solution. To complete the registration process, the user has to prove her identity for example by means of a passport or an existing eID. In order to allow for a flexible setup of the registration process and to cover a broad range of legal and organizational requirements, the registration process has been designed to support different types of registration. These types of registration cover use cases from the e-government domain as well as use cases from related private-sector domains such as e-commerce or e-business. Furthermore, the proposed architecture is flexible enough to allow for an easy integration of further alternative registration types, in case they are required by the given use case. So far, the following four types of registration have been defined.

- **Registration via registration officer:** The identity of the user is verified face-to-face by a registration officer (RO) using official IDs, e.g., a passport or a driving license. After the verification of the user's identity, the RO manually registers the user in the proposed solution by filling the registration form with user-specific data.
- **Offline registration:** This registration type takes place in an asynchronous way. A user-data form has to be filled by an RO, after identifying the user similar to the registration type sketched above. After a validity check, the collected data has to be signed by the RO. The signed data is transmitted to the proposed solution and an activation code linked to the data is generated and passed to the user. The registration can be completed by the user at a later date using the issued activation code.
- **Self registration:** Self registration is carried out by the user herself with the help of an existing eID. While self-registration is common practice at online platforms, our solution relies on existing qualified eIDs for this purpose. The system verifies the user's identity by means of the provided eID and enables her to complete the registration afterwards on her own. An RO is not required for this type of registration, as the verification of the identity must have happened before during the activation of the existing eID.
- **Registration via trusted organization:** Many organizations have the legal requirement to identify their customers. Examples are bank institutes or universities. If a trust relationship with these organizations is established, existing identification data from these organizations can be used to register new users.

Figure 2 illustrates the general registration process of the proposed solution. The basic goal of the registration process is the creation of a Standard Identification Record (SIR) for a specific user. The SIR can be created using the four registration types sketched above. Irrespective of the applied registration type, a SIR is created which unambiguously identifies a user and provides this user basic access to the proposed solution.

Support of different types of registration allows for a very flexible setup of the registration process and covers a broad range of legal and organizational requirements regarding the registration process. This, in turn, contributes to a flexible operation of the proposed solution. This way, the proposed solution fulfills Requirement R5.

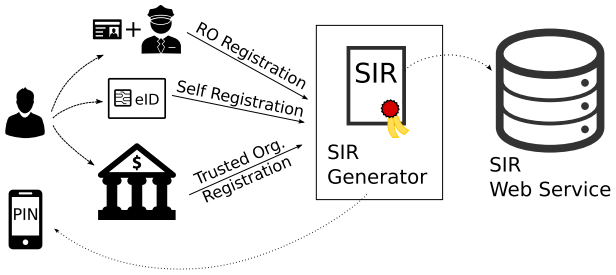


Figure 2: Registration.

B. Activation Process

After successful registration, users can run the activation process to create a new eID. For this purpose, the user needs to log-in to the proposed solution. This is only possible, if a valid SIR is available for the user (i.e., if the user has successfully completed the registration process) or if the user has already created an eID during an earlier activation process. In the former case, the user is unambiguously identified by means of the SIR. In the latter case, the user can log-in using the already created eID.

After a successful log-in, the user can create a new eID. For this purpose, the user is asked to fill the activation form. In general, the proposed solution supports multiple eIDs for each user. Therefore, the activation process can be run multiple times by each user. Each created eID can be managed separately. This enables users to have eIDs for different purposes, e.g., private and official affairs. During each activation process, a new cryptographic key pair is created for the user. This key pair can be used for subsequent signature-creation processes. Additionally, a certificate is issued to bind the user's identity to the created key. For each created eID, specific authentication data need to be defined by the user including a secret signature password (which will be verified against a regular expression pattern defined by the system administrator) and a mobile-phone number. The user has to prove possession of the specified mobile phone. This is achieved by means of OTPs that are sent to the user through an OTP Gateway.

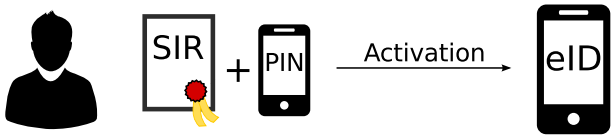


Figure 3: Activation.

In addition to the created key pair and the defined authentication data, also identity-related information, like full name and birth date, is assigned to the newly created eID. This information is obtained by the Person Register. The Person Register is a component that connects to an external database containing potential users of the service. Depending on the deployment and application scenario, this can be an existing official database like a central register of residence maintained by a public authority, an existing domain-specific database like the database of employees of a private-sector company, or a database specifically operated for this service that grows with every new registration. After fetching required identity-related information from the relevant database, the Person Register

returns a signed data structure that contains the unique eID of the applicant and also the public key of the created signature key-pair. This way, it is possible to link a signature to a person for means of identification without the need to embed the unique eID directly in the signing certificate. By clearly separating eID functionality from e-signature functionality the users' privacy is assured. A similar concept has already been successfully applied in existing national eID solutions [20].

After completion of the activation process, a new eID has been created. This eID comprises signed identity-related information and a key pair (and certificate) for the creation of electronic signatures. Additionally, a secret password has been chosen and a mobile-phone number has been registered for the created eID, which are required during subsequent usage processes. The basic principle behind the activation process is illustrated in Figure 3.

C. Usage Process

After the successful completion of the activation process, the user can use the created eID to securely and conveniently authenticate at services and to create electronic signatures. To create an electronic signature or to authenticate, the user has to enter her phone number and signature password. If the data provided by the user can be verified, an OTP is sent to her mobile phone in order to verify possession of the mobile phone. If the user can prove possession of the mobile phone by entering the OTP, the requested signature creation is performed. The main concept behind the usage process is shown in Figure 4.

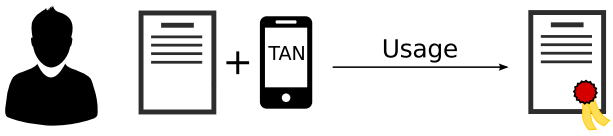


Figure 4: Usage.

VI. Implementation

Based on the proposed architecture and the defined processes, we have implemented a prototype to evaluate and demonstrate the applicability of our solution. This prototype has been named ServerBKU. The ServerBKU represents a server-based eID and e-signature solution. In the following subsections, we elaborate on the technologies used to realize the ServerBKU and discuss in detail the implementation of the ServerBKU's main processes.

A. Choice of Technologies

We have built our prototype implementation on a set of well-known and production-ready Java-based frameworks and libraries. This way, an efficient development process has been achieved and the probability of implementation errors has been minimized. Furthermore, reliance on appropriate frameworks assures that the prototype meets the requirements defined in Section III. Employed development frameworks, used libraries, and their underlying technologies are briefly introduced in this section.

The foundation of all implemented modules is the Spring Framework [21], which supports the development of modular and flexible software solutions. The basic underlying approach, followed by the Spring framework that enables a flexible design, is called dependency injection (DI). Following this approach, the dependencies of the various components are wired, i.e., injected, during runtime by the so-called inversion of control (IoC) container, a core component of the Spring framework. During development, concrete functionality, e.g., the OTP gateway, is abstracted by interfaces or base classes. Concrete implementations of the abstracted functionality are selected by configuring the IoC container. In the case of the OTP gateway, for instance, a special SMS-gateway implementation has been selected to implement the functionality of the OTP gateway interface. The flexible and easy selection of concrete implementations for abstract functionality enables a loose coupling of modules and allows software to be tailored to the specific needs of the use-case at hand. The loose coupling of modules also facilitates a test-driven development, as a single component can easily be tested without many dependencies. The concepts of DI and IoC are actually no unique features of the Spring framework. The Spring framework just implemented these concepts from the very beginning in 2002 in order to enable the development of flexible software.

Apart from the concepts DI and IoC, the Spring framework also provides templating mechanisms for various common tasks. This minimizes the amount of boilerplate code, which in turn reduces the chances of copy-and-paste errors and keeps the source code slim and readable. A prominent example of templates provided by the Spring framework is the Hibernate template. Hibernate [22] is an object-relational mapping (ORM) library, i.e., entries of relational databases are mapped to Java objects and vice versa. This way, most of the specifics of an underlying database can be abstracted by Hibernate. This enables an adoption of databases to the needs of certain deployment scenarios. By relying on the Spring framework, our prototype implementation, i.e., the ServerBKU, achieves a sufficient level of flexibility as demanded by the requirements defined in Section III.

In order to further improve the flexibility of the ServerBKU, a suitable proxy mechanism has been selected. This mechanism enables data exchange between different modules of the ServerBKU. For this purpose, the Java messaging service (JMS) API has been the technology of choice. Using this technology, the actual instance of an interface may run transparently on a different host. This way, it is possible to run the complete stack on a single machine or distribute the components over several servers. Apache ActiveMQ [23] has been chosen as implementation of the JMS API. Apache ActiveMQ supports out of the box redundancy and load balancing mechanisms. Furthermore, all exchanged messages can be protected via TLS secured channels. For this purpose, the IAIK iSaSiLk library has been used, which provides an extensible and highly configurable implementation of SSL 2.0 and 3.0 and TLS 1.0 and 1.1.

Libraries provided by IAIK [24] have also been employed to implement required cryptographic operations. Concretely, the IAIK provider for the Java Cryptography Extension (IAIK JCE) has been used to implement relevant functionality. Furthermore, the ServerBKU relies on the IAIK ECCelerate library to implement functions related to elliptic curve cryptography. To access the hardware security module (HSM), the ServerBKU uses the IAIK PKCS#11 Provider and Wrapper. The wrapper provides the Java Native Interface (JNI) to the hardware-dependent PKCS#11 library, while the PKCS#11 Provider implements a JCE provider for a specific hardware

module. You can see an overview of the technologies used in Table I on page 10.

Besides appropriate development frameworks and cryptographic libraries, also a suitable technology to implement the required OTP gateway has finally been selected. Concretely, the ServerBKU has been defined to use transaction numbers (TAN), which are generated randomly, delivered via an SMS gateway to cover the functionality of OTPs and the OTP gateway. The employed SMS gateway operator provides an proprietary interface that enables the delivery of SMS messages via HTTP POST.

To assure the security of the ServerBKU, appropriate technologies have been chosen to assess our implementation by means of systematic security analyses. To follow an approved approach, the ServerBKU has been evaluated regarding the most recent critical risks according to OWASP [25]. Risks and flaws proposed by OWASP to be investigated are for example different types of code injection, Cross Site Scripting (XSS) or Cross Site Request Forgery (CSRF) amongst others. Analyses have been carried out using a white-box testing approach, as this method reveals most implementation errors. Several tools exist that facilitate such tests. Examples are Burp Suite [26] and several useful browser plugins that for instance allow the editing of cookies. Following the white-box approach allows the auditor having knowledge of the internal structure of the project, like the knowledge of libraries and frameworks in use, as well as having access to the source code. This way, the ServerBKU has been systematically and reliably assessed in terms of security.

B. Realization of Processes

Based on selected technologies, development frameworks, and libraries, the three processes defined in Section V have been implemented. The implementation of these processes is described in detail in the following subsections.

1) Registration Process: In this step, the user has to prove her identity. Our implementation supports all types of registration defined in Section V. In a traditional setup, registration happens at the office of the RO. For this scenario, our implementation provides a web-based UI, through which the RO can register the user in the system by entering user data after identity verification. This UI is shown in Figure 5.

However, in some situations it might be beneficial for the RO to travel from user to user. This requires means to carry out asynchronous offline registration, as access to the ServerBKU is potentially not available at the user's place. To support this type of registration, the ServerBKU supports registration of users via SIRs created offline. A SIR contains information to identify a person, information about the ID used to verify the identity of the user, a binding towards a hardware token, i.e., a mobile phone for the use case at hand, and the electronic signature of a RO. Alternatively, a SIR can also be signed by a trusted partner, e.g., a bank or a university. This corresponds to the fourth type of registration listed in Section V. SIRs can be created from data entered by the RO (or trusted partner) or by the user using additional software. During the creation of the SIR, an activation code is generated and delivered to the user, cf. Figure 2.

Created SIRs must be sent to the ServerBKU's external SIR web service component via SOAP. The SIR webservice verifies the validity of a provided SIR by means of its electronic signature. If this verification succeeds, the SIR is stored in the user database of the ServerBKU. The user can use the stored SIR together with the activation code at a later date to start the activation process. The ServerBKU supports different front-ends that enable this type of registration. Initially, we developed a simple, yet comprehensive, stand-alone application based on Spring MVC. This application can be used on mobile devices in case of traveling ROs and supports the RO in creating SIRs (Figure 5). Furthermore, we developed an interface component that enables a traveling RO to take a picture of the ID of the user to be registered. The required data is extracted from this picture using optical character recognition (OCR). From these data, the required SIR is finally created. By supporting differnet means to create SIRs offline, the ServerBKU facilitates the offline registration of users.

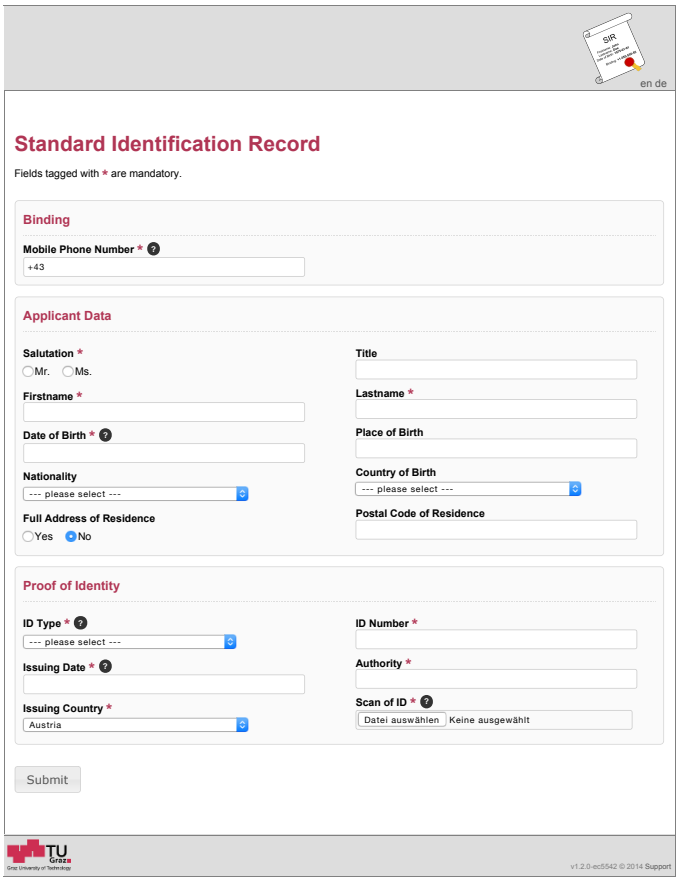


Figure 5: Offline Registration.

To cover the last registration type, the ServerBKU provides a UI for the user. This UI is similar to the one developed for registrations vio ROs. It allows the user to carry out a self-registration in case she has already a trusted eID, e.g., smart card. As the user is identified and authenticated by means of this eID, no RO is necessary to complete the registration process.

2) Activation Process: In this process, the user creates and activates a new mobile eID. The activation process offers again a web-based interface. It has been developed using Java Server Faces (JSF) 2.1 [27] and Primefaces [28] for the frontend. The decision to use a different technology to create the UI is based on the rich set of UI components that is part of Primefaces. This facilitates development of a flexible, easy to

use, role/permission-based interface in a short amount of time.

If the registration was performed in the classical way including an RO or as self-registration, the activation process starts automatically after registration. A pre-registered user can start the activation any time and independent of the registration process by submitting the received activation code and her telephone number to a specific URL. This way, available user data is automatically pre-populated as far as possible in the provided activation form by extracting the corresponding data from the SIR received from the database. Additionally, required data such as a signature password and a revocation password have to be entered by the user into the activation form.

As users may activate an arbitrary number of mobile eIDs for each phone number, activated eIDs have to be distinguishable by the system. This is achieved by the SHA-512 hash of the phone number and the signature password that have been selected by the user. Consequently, passwords have to be unique per telephone number. As the phone number is usually constant, a unique password has to be chosen for each eID. To verify the user's phone number and the possession of the device, a random OTP is generated, sent to the user's phone, and queried at the web interface. If the user enters a wrong OTP too often or if the code has expired, the activation process is aborted. The length and appearance (e.g., numeric, alphanumeric, etc) of this OTP as well as the number of trials and the time of validity is configurable. The user has also the possibility to resend the OTP a configured number of times in case the message gets lost on its way.

After the user has proven possession of the mobile phone, a signing key-pair for the user is created in the HSM. The private key is then wrapped by and exported from the HSM and securely, i.e., encrypted, stored in the ServerBKU's database. For details on the encryption scheme see below. Additionally, a certificate signing request (CSR) is generated. The public key is extracted from the CSR and sent to the Person Register together with additional data such as name and date of birth, which are required to identify the user in the Person Register. The Person Register returns a signed data structure that contains the unique eID of the user and the public key of the created signature key-pair. The returned signed data structure is, again encrypted, stored in the ServerBKU's database. Subsequently, an end-user certificate is requested from the CA using the already created CSR. The obtained certificate is stored together with the private key and the created eID data in the database.

The encryption of stored user data is based on a secret signature password, which the applicant chooses during the activation process. The ServerBKU relies on a hybrid encryption scheme as suggested by Orthacker et al. [15]. Here, the user has an additional encryption key-pair ($K_{\text{enc}}^{\text{pub}}$ and $K_{\text{enc}}^{\text{priv}}$), which is generated alongside the signing key-pair. The private key is then encrypted ($EK_{\text{enc}}^{\text{priv}}$) under the users signature password (PW_{sig}) and stored in the database. This happens only once during the activation phase.

$$SK_{\text{PW}} = \text{derive}(PW_{\text{sig}}) \tag{1a}$$
$$EK_{\text{enc}}^{\text{priv}} = \text{encrypt}(K_{\text{enc}}^{\text{priv}}, SK_{\text{PW}}) \tag{1b}$$

To encrypt a plain message M, a random symmetric key SK_{rand} is generated. This random secret key has to be encrypted for the user using her public encryption key (2b) and stored together with the cipher text (2a). However, this does not involve data from (1a) and (1b).

$$EM = \text{encrypt}(M, SK_{\text{rand}}) \tag{2a}$$
$$ESK = \text{encrypt}(SK_{\text{rand}}, K_{\text{enc}}^{\text{pub}}) \tag{2b}$$

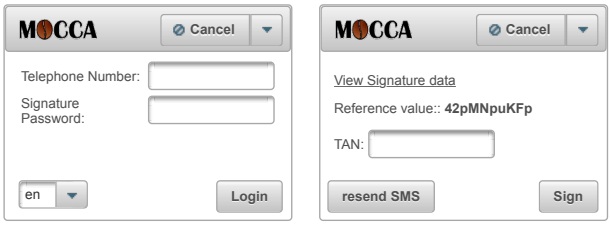
This enables encryptions of data on behalf of the user without knowledge of the user's signature password. The decryption, however, requires the consent of the user, which she gives by providing the signature password (3a).

$$SK_{\text{PW}} = \text{derive}(PW_{\text{sig}}) \tag{3a}$$
$$K_{\text{enc}}^{\text{priv}} = \text{decrypt}(EK_{\text{enc}}^{\text{priv}}, SK_{\text{PW}}) \tag{3b}$$
$$SK_{\text{rand}} = \text{decrypt}(ESK, K_{\text{enc}}^{\text{priv}}) \tag{3c}$$
$$M = \text{decrypt}(EM, SK_{\text{rand}}) \tag{3d}$$

After the generated certificate has been stored in the Server BKU's database, the user gets a notification per e-mail that the activation of the eID was successful. This finally completes the activation process.

Apart from the actual activation process, the implemented user interfaces also provide additional functionality. For instance, an interface has been implemented for ROs to perform activations on behalf of someone else as a usability feature. Furthermore, an interfaces is provided for each user that facilitates the management of eIDs, both for the user and also for a support team. This interface is shown in Figure 7. Finally, an administration UI has been developed that allows the definition and assignment of roles.

3) Usage Process: The usage process has been developed alongside the activation process and therefore is built on the same technologies, i.e., JSF [27] and Primefaces [28]. The interfaces are reduced to the bare minimum required for authenticating users and authorizing the creation of electronic signatures. This facilitates an easy integration of the Server-BKU into arbitrary third-party applications. The two forms that are used during the user-authentication process are shown in Figure 6.



(a) Login (b) TAN Verification

Figure 6: Interface of the Usage Process.

The signature-creation process starts with the receipt of an appropriate HTTP POST request at the web interface provided by the ServerBKU. The system returns the form shown in Figure 6(a), where the signer has to provide her phone number and signature password. The signature password is used to decrypt a private key that is part of the hybrid encryption used to securely store user-related data. Thus, neither the decryption

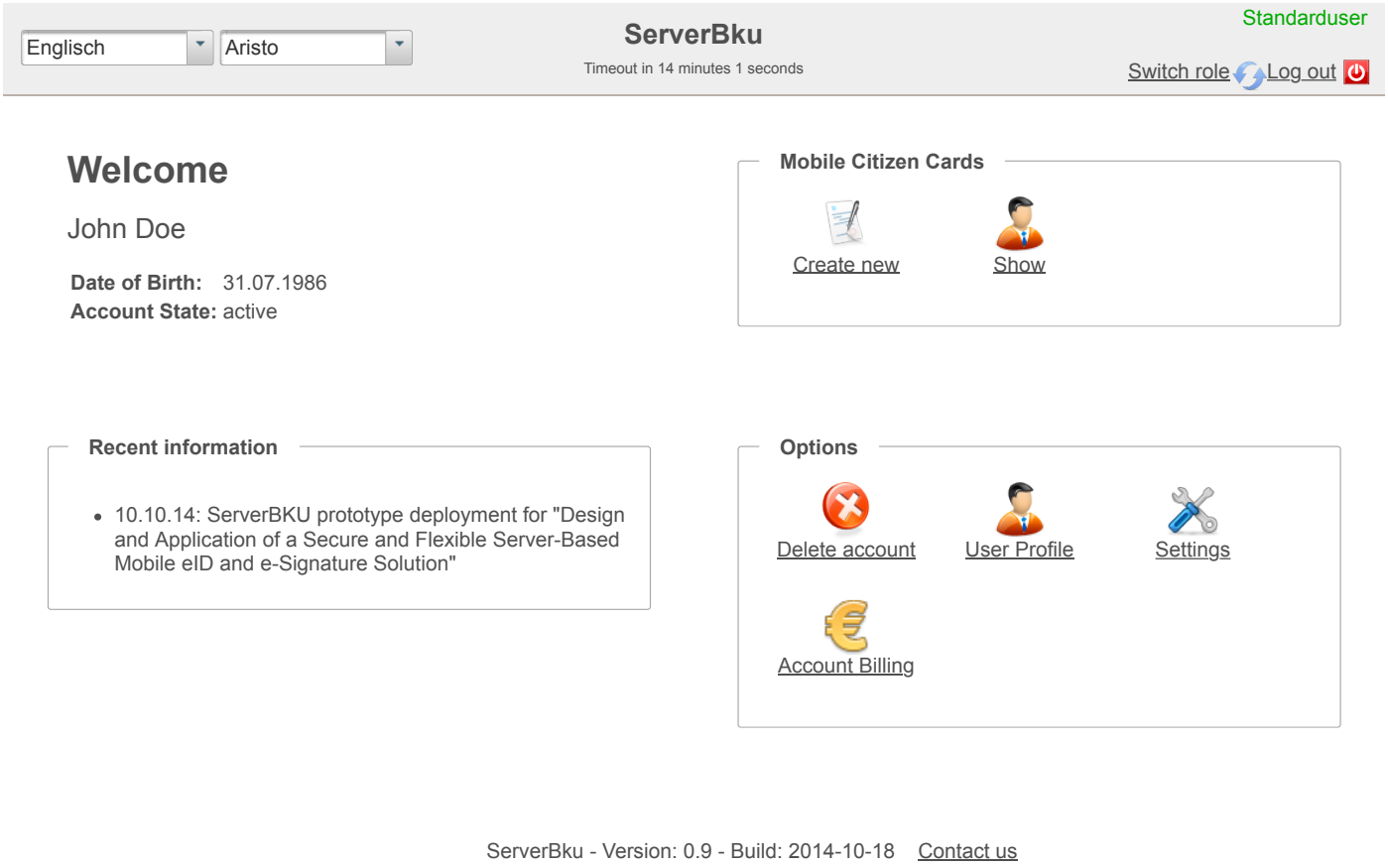


Figure 7: Activation Management.

of the user's signature key has to take place before the two-factor authentication is complete, nor must the signature password be stored in a session. In order to prevent brute force attacks, the account for a given phone number gets locked a configurable period of time if too many unsuccessful log-in attempts are recognized.

If the user authentication was successful, two random values are generated: the OTP, i.e., the TAN, and a reference value, which is displayed in the TAN verification form (Figure 6(b)) and in the SMS that is used to deliver the TAN. This way, a link is provided between the TAN and the current session. Next, the service sends the TAN via the OTP gateway to the user's mobile phone in order to verify its possession.

After verifying the reference value received by SMS against the reference value displayed in the TAN verification form, the user enters the received TAN into the form. The form also provides a link to display the signature data. This enables the user to check the data to be signed prior to authorizing the signature creation. If the user has been successfully authenticated, the user data is read from the database and decrypted using the user's private key of the hybrid encryption scheme. Then, the still-wrapped private key of the signing key-pair is loaded into the HSM where it is unwrapped. Thus, the users private signing key is never accessible in a usable form outside the HSM. Finally, the unwrapped key is used inside the HSM to create an electronic signature on behalf of the user. After successful completion of the signature-creation process, the unwrapped key is discarded and the created signature is returned to the requesting entity.

VII. Evaluation

The ServerBKU shows that the server-based signature solution proposed in this article can be implemented in practice. To further evaluate its applicability in real-world use cases, we have deployed the ServerBKU in-house and linked it to an already existing application. We elaborate on this deployment and on the evaluation of the ServerBKU in this section. For this purpose, we first introduce the in-house application Timesheep, which has been used to evaluate the ServerBKU. We then show how the ServerBKU has been integrated into Timesheep to evaluate its applicability.

A. Timesheep

In the past, our organization used simple Excel sheets to record efforts, i.e., working hours, for employees and projects. Each employee had to fill out an Excel sheet with the efforts he spent on assigned projects. These Excel sheets were printed and had to be signed by the user by hand. After signing, the Excel sheets were forwarded to the group leaders. In the last step, responsible group leaders had to sign the Excel sheets themselves, in order to approve them. Signed Excel sheets were archived for project calculations. This process was cumbersome for several reasons including the following ones.

- Employees often forgot to fill out their Excel sheets in time and had to be reminded frequently.

- Employees sometimes forgot to print and sign Excel sheets, which caused delays in project calculations.
- Excel sheets had to be maintained and forwarded to group leaders manually, representing a potential source of error.
- When out of office, group leaders were not able to sign Excel sheets resulting in delays in project calculations.

To overcome these problems, our organization now uses a web-based time tracking tool called *Timesheep*. It tracks the efforts, i.e., working hours, for each employee and project. Timesheep runs on a virtual machine, which is only accessible from our internal network or through a Virtual Private Network (VPN) connection. Timesheep runs on similar technologies like the ServerBKU. It uses the Spring Framework [21] as a basis for modular and flexible development. Hibernate [22] is used to make the implementation independent from the underlying database. In addition, Spring Roo [29] has been used for fast prototyping. Spring Roo has been configured to use Spring Web MVC [30] as the web-rendering framework. To extend Spring Web MVC's tagx components, Prime UI [31], handsontable [32] and vis.js [33] have been used. You can see an overview of the technologies used in Table I.

TABLE I: Choices of Technologies Overview

Technologies / Application	ServerBKU	timesheep
Java	X	X
Spring Framework	X	X
Spring Roo		X
Spring WEB MVC		X
handsontable		X
vis.js		X
Primefaces	X	
Prime UI		X
Hibernate	X	X
Apache ActiveMQ	X	
IAIK iSaSiLk	X	
IAIK JCE	X	X
IAIK Eccelerate	X	
IAIK PKCS#11 Provider and Wrapper	X	

Timesheep defines several roles within our organization, in order to model required functionality:

- **User:** The first role is the role of normal users, i.e., employees. Users must have an easy access to Timesheep to track their efforts. This is achieved by providing a simple web-based interface, which can be accessed by using any common web browser.
- **Group Leader:** The second role is the role of group leaders. Group leaders must be able to access tracked efforts of employees assigned to their group, check these efforts, and approve, i.e., sign, them. Furthermore, the group leaders must be able to plan projects based on the budget of a project and on the efforts assigned users can raise until the project deadline.
- **Administrator:** The third role is the role of the administrator. Timesheep was developed to minimize the efforts administrators have to do. New employees are automatically added to the system with the necessary information for Timesheep to work. This is achieved by linking Timesheep to our in-house domain log-in system. This approach has also been followed by the ServerBKU, in order to avoid double registrations and to achieve maximum comfort. Thus, employees were able to log-in to Timesheep and to the ServerBKU with their domain log-in name without an additional registration process.
- **Financial Department:** The fourth role is the role of the financial department. Every project has milestones and a defined end date. At every milestone and end date, the current costs of the project must be calculated and submitted. Every submission is double checked by external financial auditors. Therefore, the auditors need the efforts done by each user on each project. After the project calculations are accepted by the auditors, tracked efforts must be protected against subsequent changes.

Based on these roles, Timesheep provides all required functionality to facilitate the tracking of efforts and the generation of time sheets. As all data are collected by one central web application and stored in one central database, required processes can be automated and delays caused by the manual tracking of efforts and processing of Excel sheets can be eliminated.

Although Timesheep has significantly improved the tracking of efforts in our organization, room for improvement could still be identified. The main drawback of Timesheep was its reliance on handwritten signatures. Even though efforts are tracked in electronic form and stored centrally in a database, employees and group leaders still had to sign generated time sheets per hand. Even though means to sign documents electronically exist, these means were not integrated into Timesheep.

To overcome this issue, we have integrated the ServerBKU into Timesheep. This way, Timesheep has been enhanced by means to electronically sign generated time sheets. Furthermore, this integration evaluates the practical applicability of the ServerBKU. The integration of the ServerBKU into Timesheep is discussed in the following section.

B. Combining Timesheep and ServerBKU

After all required efforts have been entered by the user and approved by the group leader, Timesheep creates a PDF-based time sheet called *monthly timesheet*. The monthly timesheet has to be signed by both the user and the responsible group leader. By integrating the ServerBKU, this signing process has been improved in terms of efficiency. For this purpose, we have integrated the ServerBKU's signature process seamlessly into Timesheep in order to achieve the best usability possible.

Timesheep is organized into multiple modules as shown in Figure 8. Each module fulfills a specific purpose for our organization.

- **Time Tracking Module:** This module offers a web-based interface for users to store their efforts.
- **Planning Module:** This module provides group leaders project planning and budget estimation based on user's contracts and employment status, and on project deadlines. Calculated values, e.g., hours, budgets, etc., are the so called *target* values.
- **Monthly Timesheet Module:** This module is responsible for generating and managing monthly timesheets. This module is also responsible for handling the signature process with the ServerBKU.
- **Financial Module:** This module offers our financial department the possibility to check how much a project did actually cost. These values are the so called *actual* values and may differ from the *target* values

defined by the Planning Module. Actual values will be submitted to external financial auditors.

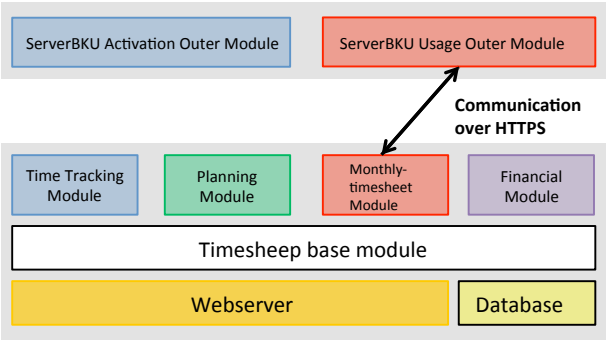


Figure 8: Timesheep Modules.

Figure 8 shows that the monthly timesheet module is responsible for the generation and management of timesheets. Hence, this module needs to be enhanced, in order to integrate the ServerBKU into the timesheet management and signing process. The ServerBKU-based signature process of a timesheet is shown in Figure 9. This figure shows how the monthly timesheet module interacts with components of the ServerBKU to electronically sign generated time sheets.

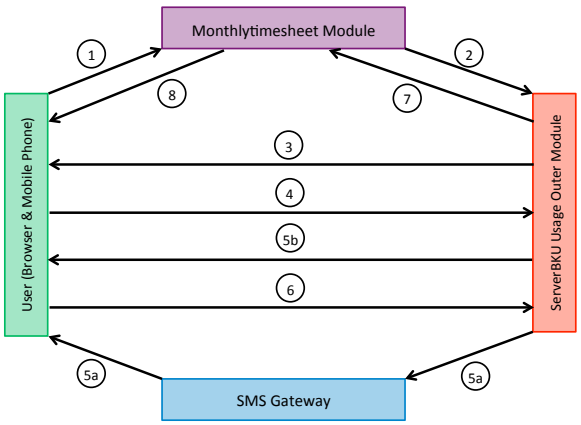


Figure 9: Work Flow of the Signature Process.

In the initial Step (1), the user starts the signature-creation process by clicking a button in his browser. A new browser window opens, in which all further communication between the ServerBKU and the user takes place. In Step (2), the monthly timesheet to be signed is sent to the ServerBKU. In Step (3), the ServerBKU prepares a PDF Advanced Electronic Signature (PAdES) and displays an authentication form to the user as shown in Figure 6(a). After that step, the user enters her phone number and signature password (Step (4)). In the next step, the ServerBKU sends a generated TAN to the user's mobile phone (Step (5a)) and displays the TAN verification form (Step (5b)). Next (Step (6)), the user enters the received TAN in the TAN verification form as shown in Figure 6(b). If the TAN provided by the user was successfully verified, the ServerBKU signs the monthly timesheet and sends the signature back to Timesheep (Step (7)). The browser window opened in Step (1) closes. Timesheep receives the signature, verifies it and notifies the user that the signature has been successfully verified. This is covered by Step (8). Signed monthly timesheets are stored in Timesheep's database and can be provided to external financial auditors to report efforts done by users.

By integrating the ServerBKU into our in-house application Timesheep, two goals have been reached. First, the process of signing time sheets containing tracked efforts of employees has been improved in terms of efficiency and usability. Even though the performance of the developed solution has not been systematically measured so far, related work on the usability of server-based signature solutions indicate that these solutions are advantageous in terms of usability [5]. This is also supported by first practical experiences gained with the ServerBKU-enhanced Timesheep instance. These experiences show that the integration of the ServerBKU improves the user acceptance of Timesheep and helps to reduce delays in reporting efforts to the financial department. A systematic measurement of the concrete usability and performance improvement that has been reached by integrating the ServerBKU into Timesheep is regarded as future work. Second, integration of the ServerBKU into our in-house application Timesheep shows that the proposed server-based signature solution in general and its concrete implementation ServerBKU in particular are applicable in practice and can be smoothly integrated into existing applications. Thus, the proposed server-based signature solution and the ServerBKU have been evaluated successfully.

VIII. Conclusion

In this article, we have proposed, presented, and discussed an enhanced server-based eID and e-signature solution. Based on a set of relevant requirements, we have developed an appropriate architecture for the proposed solution first. We have then carried this architecture over to a concrete implementation called ServerBKU using common state-of-the-art technologies. A test deployment of this implementation is publicly available online and can be accessed for test purposes [34]. Furthermore, the practical applicability of the ServerBKU has been evaluated by integrating it into the time-tracking tool Timesheep.

Even though the ServerBKU is ready for productive use, there are still some open issues that are regarded as future work. First, we need to gain more practical experience with our solution especially with regard to different deployment and application scenarios. Although first empirical results obtained by integrating the ServerBKU into the time-tracking tool Timesheep are promising, further experiences are required to further develop and optimize our solution. Second, we want to systematically measure the efficiency of the ServerBKU, in order to identify potential usability limitations. For instance, Single Sign-on solutions could help to reduce required user interactions and, hence, improve efficiency and usability.

While the concept of server-based eID and e-signature solutions is not completely new, the ServerBKU is the first one that is not tailored to a certain application scenario. While existing solutions such as the Austrian Mobile Phone Signature have been developed for a specific deployment scenario, the ServerBKU has been designed such that it can be easily integrated into arbitrary application and deployment scenarios. This way, the ServerBKU leverages the use of eID and e-signature functionality in arbitrary applications and

helps to improve their provided level of security. In particular, the ServerBKU offers application an attractive alternative to insecure password-based authentication schemes, which will hopefully be history in the future.

Acknowledgements: The authors have been supported by the European Commission Seventh Framework Programme through project *FutureID*, grant agreement number 318424.

REFERENCES

[1] C. Rath, S. Roth, M. Schallar, and T. Zefferer, "A secure and flexible server-based mobile eID and e-signature solution," in Proceedings of the 8th International Conference on Digital Society, ICDS 2014, Barcelona, Spain. IARIA, 2014, pp. 7 – 12.

[2] B. Ives, K. R. Walsh, and H. Schneider, "The domino effect of password reuse," Commun. ACM, vol. 47, no. 4, Apr. 2004, pp. 75–78, [accessed November, 2014]. [Online]. Available: http://doi.acm.org/10.1145/975817.975820

[3] D. Florencio and C. Herley, "A large-scale study of web password habits," in Proceedings of the 16th international conference on world wide web, ser. WWW '07. New York, NY, USA: ACM, 2007, pp. 657–666, [accessed November, 2014]. [Online]. Available: http://doi.acm.org/10.1145/1242572.1242661

[4] S. Arora, "National e-id card schemes: a european overview," Inf. Secur. Tech. Rep., vol. 13, no. 2, May 2008, pp. 46–53, [accessed November, 2014]. [Online]. Available: http://dx.doi.org/10.1016/j.istr.2008.08.002

[5] T. Zefferer and V. Krnjic, "Usability evaluation of electronic signature based e-government solutions," in Proceedings of the IADIS International Conference WWW/INTERNET 2012, pp. 227 – 234.

[6] A. Ruiz-Martinez, D. Sanchez-Martinez, M. Martinez-Montesinos, and A. F. Gomez-Skarmeta, "A survey of electronic signature solutions in mobile devices," JTAER, vol. 2, no. 3, 2007, pp. 94–109. [Online]. Available: http://dblp.uni-trier.de/db/journals/jtaer/jtaer2.html#Ruiz-MartinezSMG07

[7] "Handy signatur und buergerkarte," 2014, [retrieved: November, 2014]. [Online]. Available: http://www.buergerkarte.at/

[8] "Estonia eID," 2014, [accessed November, 2014]. [Online]. Available: http://www.id.ee/?lang=en

[9] "Belgium eID," 2014, [accessed November, 2014]. [Online]. Available: http://eid.belgium.be/en/

[10] "Spanish eID," 2014, [accessed November, 2014]. [Online]. Available: http://www.dnielectronico.es/

[11] European Parliament and Council, "Directive 1999/93/ec on a community framework for electronic signatures," December 1999.

[12] 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. [Online]. Available: http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=COM:2012:0238:FIN:EN:PDF

[13] E. Pisko, "Mobile electronic signatures: progression from mobile service to mobile application unit," in ICMB. IEEE Computer Society, 2007, p. 6. [Online]. Available: http://dblp.uni-trier.de/db/conf/icmb/icmb2007.html#Pisko07

[14] P. Teufl, T. Zefferer, C. Wörgötter, A. Oprisnik, and D. Hein, "Android - on-device detection of sms catchers and sniffers," in International Conference on Privacy & Security in Mobile Systems, 2014, in press.

[15] C. Orthacker, M. Centner, and C. Kittl, "Qualified mobile server signature," in Security and Privacy – Silver Linings in the Cloud, ser. IFIP Advances in Information and Communication Technology, K. Rannenberg, V. Varadharajan, and C. Weber, Eds., vol. 330. Springer Berlin Heidelberg, 2010, p. 103–111. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-15257-3_10

[16] "Estonia mobile eID," 2014, [accessed November, 2014]. [Online]. Available: http://mobiil.id.ee/

[17] "Norway eID," 2014, [accessed November, 2014]. [Online]. Available: https://www.bankid.no/

[18] "Austrian Handy Signatur," 2014, [accessed November, 2014]. [Online]. Available: https://www.handy-signatur.at/

[19] K. Stranacher, A. Tauber, T. Zefferer, and B. Zwattendorfer, The austrian identity ecosystem - an e-government experience book title: architectures and protocols for secure information technology, ser. Advances in Information Security, Privacy, and Ethics (AISPE). Antonio Ruiz Martinez, Rafael Marin-Lopez, Fernando Pereniguez-Garcia, 2013, pp. 288 – 309.

[20] H. Leitold, A. Hollosi, and R. Posch, "Security architecture of the austrian citizen card concept," in Proceedings of 18th Annual Computer Security Applications Conference (ACSAC'2002), Las Vegas, 9-13 December 2002. pp. 391-400, IEEE Computer Society, ISBN 0-7695-1828-1, ISSN 1063-9527., 2002.

[21] "Spring Framework," 2014, [accessed November, 2014]. [Online]. Available: http://projects.spring.io/spring-framework/

[22] "Hibernate," 2014, [accessed November, 2014]. [Online]. Available: http://hibernate.org/

[23] "Apache Active MQ," 2014, [accessed November, 2014]. [Online]. Available: http://activemq.apache.org/

[24] "IAIK JCE," 2014, [accessed November, 2014]. [Online]. Available: http://jce.iaik.tugraz.at/

[25] The Open Web Application Security Project, "Owasp top 10 - 2013 the ten most critical web application security risks," 2013, [accessed November, 2014]. [Online]. Available: https://www.owasp.org/index.php/Top_10_2013

[26] PortSwigger Ltd, "Burp suite," [accessed November, 2014]. [Online]. Available: http://portswigger.net/burp/

[27] "Java Server Faces," 2014, [accessed November, 2014]. [Online]. Available: https://javaserverfaces.java.net/

[28] "Primefaces," 2014, [accessed November, 2014]. [Online]. Available: http://www.primefaces.org/

[29] "Spring Roo," 2014, [accessed November, 2014]. [Online]. Available: http://projects.spring.io/spring-roo/

[30] "Spring Web MVC," 2014, [accessed November, 2014]. [Online]. Available: http://docs.spring.io/spring/docs/3.2.x/spring-framework-reference/html/mvc.html

[31] "Prime UI," 2014, [accessed November, 2014]. [Online]. Available: http://www.primefaces.org/primeui/

[32] "handsontable," 2014, [accessed November, 2014]. [Online]. Available: http://handsontable.com/

[33] "vis.js," 2014, [accessed November, 2014]. [Online]. Available: http://visjs.org/

[34] "ServerBKU prototype deployment," 2014, [retrieved: November, 2014]. [Online]. Available: https://pheasant.iaik.tugraz.at:8443/Registration/