Towards Transactional Electronic Services on Mobile End-user Devices
A Sustainable Architecture for Mobile Signature Solutions

Thomas Zefferer
Institute for Applied Information Processing and Communications (IAIK), Graz University of Technology, Inffeldgasse 16a, 8010 Graz, Austria

Keywords: Mobile Signature Solution, Mobile Devices, Assessment, Architecture, Security, Usability.

Abstract: In the European Union, transactional e-services from security-critical domains such as e-government or e-banking often require users to create legally binding electronic signatures. Currently available solutions, which enable users to create such signatures, have usually been designed for classical end-user devices such as desktop computers or laptops and cannot be applied on mobile end-user devices. This complicates a use of transactional e-services on mobile devices and excludes a growing number of users, who prefer mobile access to services. To address this problem, this paper develops a mobile signature solution that can be applied on mobile end-user devices. Possible architectures for this solution are systematically derived from a generic model first. The best alternative is subsequently determined by means of systematic assessments. This finally yields a technology-agnostic mobile signature solution that can be used as a basis for concrete implementations. By keeping the proposed solution on a rather abstract level, its validity is assured, even if available mobile technologies and the current state of the art change. This way, the proposed solution represents a sustainable basis for future signature solutions and paves the way for transactional e-services on mobile end-user devices.

1 INTRODUCTION

Electronic signatures have evolved to a fundamental building block of transactional electronic services (e-services). Relying on asymmetric cryptographic methods such as RSA (Rivest et al., 1978) or ECDSA (ANSI, 2005), electronic signatures provide integrity, authenticity, and non-repudiation. Electronic signatures are for instance frequently used in the e-government domain to obtain written consent from remote users. Furthermore, they are also employed by e-banking solutions to facilitate a remote authorization of financial transactions. The potential of electronic signatures has been especially recognized in the European Union (EU), where EU laws such as the EU Signature Directive (The European Parliament and the Council of the European Union, 1999) or the EU eIDAS Regulation (The European Parliament and the Council of the European Union, 2014) provide a legal foundation for electronic signatures. Concretely, these laws define the concept of Qualified Electronic Signatures (QESs). QESs represent a special class of electronic signatures and are defined to be legally equivalent to handwritten signatures. Therefore, QESs are especially relevant for transactional e-services that require legally binding written consent from remote users. The relevance of QESs raises the need for signature solutions that enable users to create legally binding electronic signatures in online procedures. During the past years, such solutions have been developed in various European countries. Examples are smart card based solutions, which have for instance been introduced and deployed in Austria (Leitold et al., 2002), Belgium (Fairchild and de Vuyst, 2012), or Portugal (Agência para a Modernização Administrativa, 2015). Other signature solutions available in Europe enable users to create QESs using their mobile phones. Examples are the Austrian Mobile Phone Signature (A-Trust, 2015) or the Estonian Mobiil-ID (ID.ee, 2015). During the past years, available signature solutions have facilitated transactional e-services from various domains.

For a long time, transactional e-services have been developed for classical end-user devices such as desktop computers and laptops. Accordingly, existing signature solutions are tailored to the characteristics of these devices as well. This applies to smart card based solutions as well as to approaches relying on the user's mobile phone. Existing signature solutions implicitly assume that the user accesses e-services with

Zefferer T..
Towards Transactional Electronic Services on Mobile End-user Devices - A Sustainable Architecture for Mobile Signature Solutions.
DOI: 10.5220/0005482405860597
In *Proceedings of the 11th International Conference on Web Information Systems and Technologies* (WEBIST-2015), pages 586-597
ISBN: 978-989-758-106-9

a desktop computer or laptop and in addition makes use of a smart card or a mobile phone to create required QESs. Unfortunately, this assumption is not valid any longer. During the past few years, desktop computers and laptops have been gradually replaced by smartphones and related mobile end-user devices. This raises various challenges for e-service, which need to be prepared for access by mobile devices. In particular, this also applies to existing signature solutions for the creation of QESs. Being tailored to the characteristics of classical end-user devices, these solutions usually cannot be applied on smartphones and other mobile end-user devices. For instance, mobile devices usually lack support for card-reading devices, which are a prerequisite for smart card based solutions. Similarly, signature solutions relying on mobile phones usually cannot be applied on smartphones or tablet computers either, as their underlying security concepts have been designed for scenarios, in which the mobile phone is used as an additional device and is solely used during the creation of QESs. The inappropriateness of existing signature solutions raises the need for new solutions that enable users to create QESs on modern mobile end-user devices. Only if such solutions are provided, transactional e-services can be adapted such that they can be accessed from and used with mobile end-user devices.

To facilitate the development and provision of such a solution, this paper identifies possible architectures and assesses them with regard to relevant success factors. This way, the best architecture of signature solutions for mobile end-user devices is determined and a sustainable basis for concrete implementations is provided. To achieve this goal, a thorough methodology is followed, which is also reflected by the structure of this paper. In Section 2, requirements, success factors, and relevant target platforms are identified. From the identified requirements, an abstract model is derived in Section 3. This model is used to systematically identify possible architectures. In Section 4 and Section 5, all possible architectures are assessed in terms of the success factors identified in Section 2. This way, the most suitable architecture is determined. Conclusions are finally drawn in Section 6.

2 PRELIMINARIES

Signature solutions that enable users to create QESs on their mobile devices need to satisfy several requirements. At the same time, these solutions also need to consider relevant success factors. Finally, relevant target platforms must be identified, on which these solutions shall be applicable. As preparation for the identification of possible architectures, these aspects are discussed in this section.

2.1 Requirements

Putting the focus on the EU, relevant requirements for legally binding electronic signatures, i.e. QESs, are mainly defined in EU laws. Concretely, the EU eIDAS Regulation (The European Parliament and the Council of the European Union, 2014), which is about to replace the EU Signature Directive (The European Parliament and the Council of the European Union, 1999), defines the concept of QESs, its relation to other types of electronic signatures, and requirements that must be met by QESs. From this regulation, the following set of basic requirements for QESs can be extracted:

- **R1: Reliance on QSCD.** QESs must be created with a Qualified Signature Creation Device (QSCD). QSCDs are certified hardware devices that are able to reliably and securely store cryptographic key material and that are capable to carry out cryptographic operations using this key material. Requirements of QSCDs are defined in Annex II of the EU eIDAS Regulation (The European Parliament and the Council of the European Union, 2014). Requirements for the related concept of Secure Signature Creation Devices (SSCDs), which represent the pendant to QSCDs in the EU Signature Directive (The European Parliament and the Council of the European Union, 1999), are also defined in relevant standards such as the CEN Workshop Agreement 14169 (CEN, 2004).
- **R2: Reliance on Qualified Certificates.** According to the EU eIDAS Regulation (The European Parliament and the Council of the European Union, 2014), QESs must be based on qualified certificates. These are electronic certificates that need to satisfy several special requirements, which are defined in Annex I of the EU eIDAS Regulation (The European Parliament and the Council of the European Union, 2014). Most of these requirements define mandatory contents of qualified certificates.
- **R3: Appropriate User Authentication.** QESs are based on so-called advanced electronic signatures (AdESs), which are also defined by the EU eIDAS Regulation (The European Parliament and the Council of the European Union, 2014). According to Art. 26 of this regulation, AdESs and hence also QESs must be *'created using electronic signature creation data that the signatory*

can, with a high level of confidence, use under his sole control' (The European Parliament and the Council of the European Union, 2014). This implies that solutions for the creation of QESs must implement a reliable user authentication and authorization mechanism, in order to reliably protect the user's cryptographic key material stored in the QSCD. This usually implies application of a multi-factor authentication scheme including for example the authentication factors Knowledge and Possession.

Based on these three basic requirements, possible architectures of signature solutions for mobile end-user devices will be systematically derived in Section 3.

2.2 Success Factors

According to the followed methodology, identified possible architectures will be assessed with regard to relevant success factors. So far, there is hardly any specific related work on success factors of signature solutions for mobile devices. Hence, we obtain relevant success factors from related work on mobile applications, mobile government, and signature solutions for classical end-user devices. Concretely, we consider works by (El-Kiki and Lawrence, 2006), (Al-khamayseh et al., 2007), (El-Kiki, 2007), (Karan and Khoo, 2008), and (Al-Hadidi and Rezgui, 2009).

Although these works identify relevant success factors on different levels of abstraction, several common findings can be extracted. Concretely, the factors Security, Usability, and Feasibility are identified as key for success by most authors. We hence focus on these factors when assessing possible architectures of signature solutions for mobile end-user devices. The factors Security and Usability will be explicitly considered by conducting respective assessments in Section 4 and in Section 5, respectively. In addition, the success factor Feasibility is implicitly considered by limiting the identification of possible architectures to those being feasible on current mobile end-user devices.

2.3 Target Platforms

To assess possible architectures, characteristics of mobile platforms, i.e. mobile end-user devices and mobile operating systems, must be taken into account. Currently available mobile platforms differ in terms of inherent characteristics, provided features, capabilities, and limitations. Hence, a growing number of considered platforms increases the complexity of conducted assessments. Therefore, we limit our assessments to the two currently dominating platforms Google Android (Google, 2015) and Apple iOS (Apple, 2015). This is reasonable, as these two platforms together share more than 95% of the entire smartphone market (mobiForge, 2015).

3 POSSIBLE ARCHITECTURES

The development of signature solutions that support the creation of QESs is a challenging task, as several legal requirements, success factors, and characteristics of employed end-user devices need to be considered. In this section, possible architectures for such solutions are identified. For this purpose, an abstract model is introduced first. This model is derived from the identified requirements of QESs. Subsequently, possible architectures are derived from this abstract model.

3.1 Abstract Model

The legal basis of QESs defines three basic requirements. From these requirements, an abstract model of signature solutions can be derived. This model is intentionally kept on a technology-agnostic and implementation-independent level. This way, it can serve as basis for the systematic identification of possible architectures. The derived abstract model is shown in Figure 1. It identifies relevant components of signature solutions supporting QESs and illustrates basic interactions between these components.

The abstract model shown in Figure 1 consists of the Signatory and four top-level components, i.e. the QSCD, the Signature Creation Application (SCA), the Service Provider (SP), and the User Client (UC). The functionality of the SCA is covered by three subcomponents, i.e. the Signature Processing Component (SPC), the Signatory Authentication Component (SAC), and the DTBS Viewer (DTBSV). According to the model shown in Figure 1, a typical signature-creation process comprises the following steps:

1. The Signatory uses the UC to access a service provided by the SP.
2. The SP requires the Signatory to create a QES. Hence, the SP sends the Data-To-Be-Signed (DTBS) to the SPC.
3. The SPC forwards the DTBS to the QSCD, which is required to create the QES according to Requirement R1.
4. To cover Requirement R3, the QSCD requires the Signatory to provide valid Authentication Data

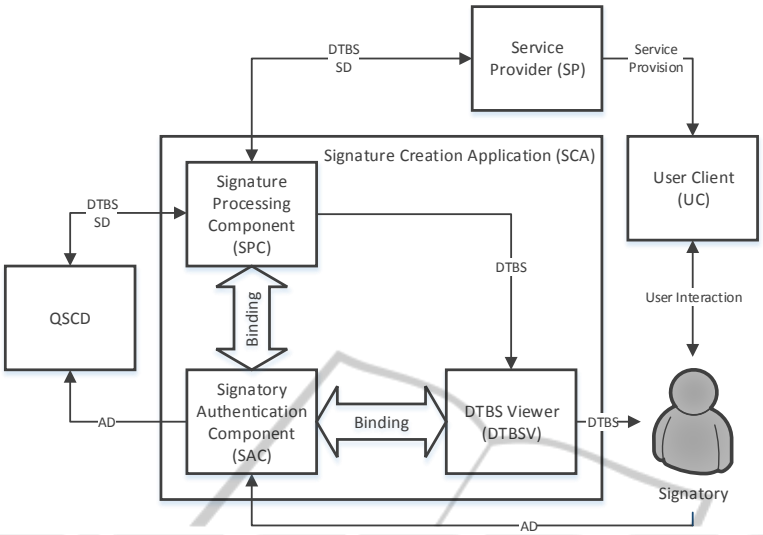


Figure 1: Abstract model.

(AD), in order to authorize the signature-creation process in the QSCD.

5. Required AD are collected by the SAC.
6. At the same time, the SPC forwards the DTBS to the DTBSV. The DTBSV displays the DTBS, so that the Signatory can check whether or not he or she wants to provide the required AD.
7. After provision of the AD, the SAC forwards the AD to the QSCD.
8. The QSCD creates the QES on the DTBS.
9. The resulting Signed Data (SD) are returned to the SPC, which forwards them to the SP.
10. The Signatory is notified of the successful signature-creation process via the UC.

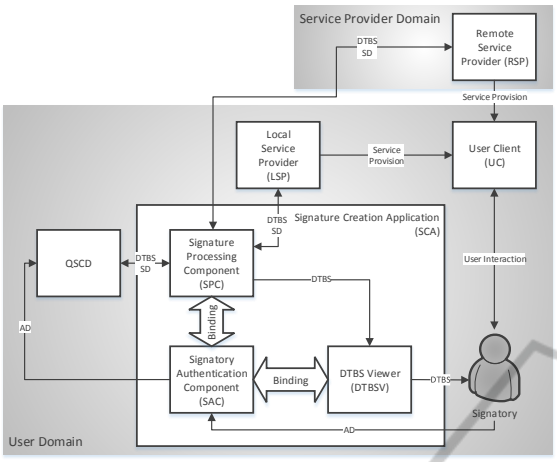
In addition to this basic process flow, two aspects need to be noted. First, there must be bindings between subcomponents of the SCA as illustrated in Figure 1. These bindings assure that the DTBS forwarded to the QSCD correspond to the DTBS displayed to the Signatory and that provided AD are used to authorize the signing of displayed DTBS only. Second, Requirement R2 is not directly covered by this abstract model. As Requirement R2 mainly concerns the structure and contents of issued signing certificates, this requirements needs to be considered during the certificate-issuing process. This mainly concerns the responsible Certification Authority (CA), which is not directly involved in signature-creation processes.

3.2 Architecture Candidates

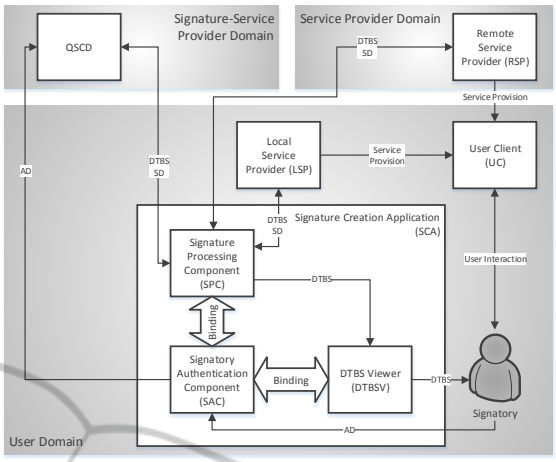
The abstract model shown in Figure 1 identifies relevant components and defines the basic process flow of a signature-creation process. Due to its abstract nature, this model is perfectly suitable to systematically derive possible architectures of signature solutions for mobile end-user devices.

To further develop the abstract model towards a concrete solution, the implementation of the identified components needs to be fixed. Aiming for a signature solution that can be applied on mobile end-user devices, identified components can—from a conceptual perspective—be implemented in two different ways. They can either be implemented locally on the mobile device or remotely in a server environment. By varying locally and remotely implemented components, different architectures can be derived. As the abstract model comprises six (sub)components, there are 64 different variations. However, only three (sub)components can be implemented remotely in practice, as all components with direct user interface to the Signatory must be implemented locally in any case. Hence, only the QSCD, the SPC, and the SP can be implemented either locally or remotely. This reduces the number of possible variations to eight. These eight variations can be subsumed to four architectures by varying the implementation of the QSCD and the SPC only. Accordingly, each of these four architectures must consider both a Local Service Provider (LSP) and a Remote Service Provider (RSP). The four resulting architectures are shown in Figure 2.

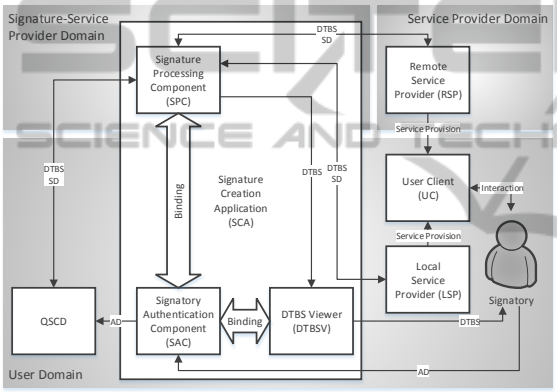
The four possible architectures, i.e. architecture candidates (ACs),all comprise the same components



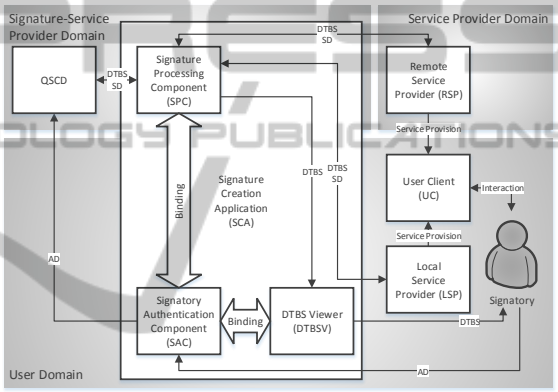
(a) Architecture Candidate A (AC A).



(b) Architecture Candidate B (AC B).



(c) Architecture Candidate C (AC C).



(d) Architecture Candidate D (AC D).

Figure 2: Architecture Candidates (ACs).

as the abstract model, from which they have been derived. However, depending on the respective AC, the components are spread over different domains. The components LSP, UC, DTBSV, and SACs are implemented in the local User Domain in any case. Similarly, the RSP is always implemented in the remote Service Provider Domain. Hence, the four ACs mainly differ regarding their implementation of the components QSCD and SPC.

As all ACs have been derived from the abstract model shown in Figure 1, they implicitly comply with the requirements identified in Section 2. Hence, they can all be used as basis for the development of signature solutions for mobile end-user devices. To determine the most appropriate approach, the four ACs are assessed with regard to the identified relevant success factors Security and Usability in the following sections.

4 SECURITY ASSESSMENT

Security is crucial for signature solutions that enable users to create legally binding QESs. These solutions must assure that created signatures cannot be forged and that access to secret cryptographic signing keys is restricted to the legitimate Signatory. The level of provided security depends on the concrete implementation, but is also heavily influenced by the underlying architecture. In this section, the four identified ACs shown in Figure 2 are assessed in order to reveal their advantages and disadvantages in terms of security.

4.1 Methodology

A thorough security assessment requires an elaborate methodology to assure meaningful results. Existing approaches such as Common Criteria (CC) based concepts (Common Criteria, 2013) are useful to as-

sess a concrete solution but are less effective for the comparison of different approaches on a pure conceptual level. Hence, we only roughly base the followed methodology on the approved concept of CC, but adapt it where necessary. Concretely, the following steps are conducted to assess and compare the four ACs in terms of security. First, assumptions are made to define the scope of the assessment. Then, relevant assets are identified and mapped to the four ACs, in order to identify protection-deserving components and communication paths. This way, the security of the four ACs can be compared on conceptual level. Finally, capabilities of the two chosen target platforms to appropriately protect protection-deserving components and communication paths are analyzed. This way, the implementation perspective is taken into account as well.

4.2 Assessment

Following the defined methodology, the scope of the conducted security assessment is defined first by making two basic assumptions. First, we assume that server-based components are secure. This is reasonable, as these components can be operated in an especially protected environment such as certified data-processing centers. Second, we assume that QSCDs are secure. This is also a valid assumption, as QSCDs need to undergo strict certification procedures.

4.2.1 Assets

After defining the security assessment's scope, relevant assets need to be identified. For the present use case, relevant assets can be extracted directly from the abstract model shown in Figure 1. Concretely, the following assets can be identified:

- **DTBS.** The DTBS must be protected from eavesdropping and unauthorized modifications by adversaries. This is crucial to assure that exactly those data defined by the SP are signed. Furthermore, the DTBS must also be protected when being displayed to the Signatory via the DTBSV, in order to enable the Signatory to check what exactly is about to be signed.
- **AD.** The confidentiality of the AD must be assured to prevent adversaries, who are able to intercept AD, from reusing them and from creating QESs on behalf of the legitimate Signatory.
- **SD.** The integrity of the SD must be guaranteed, in order to prevent adversaries from applying modifications to invalidate the created QES. Depending on the use case, the confidentiality of the SD might also be necessary.

Note that there are several additional assets that deserve protection. For instance, the Signatory's secret signing key must be kept confidential in any case. However, this key and other related assets are stored inside the QSCD, which is assumed to be secure. Hence, assets protected by the QSCD are not considered in detail for the conducted assessment.

4.2.2 Conceptual Assessment

With the help of the three identified assets, the security of the four ACs can be compared on conceptual level. For this purpose, Figure 3 lists all components, on which identified assets are potentially prone to attacks. Essentially, these are all components used by the four ACs that are not assumed to be secure. In addition, Figure 3 also lists all relevant communication paths between relevant components, on which transferred assets are potentially prone to attacks. For the sake of simplicity, several communication paths have been combined to derive six more general classes of communication paths. For instance, the communication paths between the Signatory and the DTBSV, and between the Signatory and the SAC have been combined to one communication-path class (Local Software (SW) $\leftrightarrow$ Signatory). For all listed components and communication-path classes, we have assessed their relevance for the four ACs. Concretely, we have assessed for each AC, which assets are present at which components and communication-path classes. The results of this mapping is shown in Figure 3.

Obtained results indicate that AC D is advantageous from a conceptual perspective, as it has the lowest number of components and communication-path classes, on which assets are prone to attacks.

4.2.3 Mapping to Target Platforms

The comparison on conceptual level clearly indicates AC D to be advantageous. However, the plain number of potentially vulnerable components and communication-path classes is a first indicator only. Capabilities to protect these components and communication-path classes using currently available technologies must also be taken into account. This can be achieved by analyzing related work on security features and vulnerabilities of current mobile platforms. Such analyses have for instance been provided by (Enck et al., 2009), (Rogers and Goadrich, 2012), or (Zefferer et al., 2013). From the results of these works, several interesting findings can be derived.

The security of all local components processing one or more assets depends to a large extent on security features provided by the mobile platform, on

	Architecture Candidates (ACs)											
	AC A			AC B			AC C			AC D		
	SD	*DTBS*	*AD*	*SD*	*DTBS*	*AD*	*SD*	*DTBS*	*AD*	*SD*	*DTBS*	*AD*
Components												
LSP	x	x		x	x		x	x		x	x	
SPC	x	x		x	x							
SAC			x			x			x			x
UC												
DTBSV		x			x			x			x	
Communication-Path Classes												
Local SW ↔ Local SW	x	x		x	x							
Local SW ↔ Remote SW	x	x		x	x		x	x		x	x	
Local SW ↔ Local QSCD	x	x	x						x			
Local SW ↔ Signatory			x			x			x			x
Local SW ↔ Remote QSCD				x	x	x						x
Remote SW ↔ Local QSCD							x	x				

Figure 3: Conceptual architecture comparison.

which these components are implemented. Concretely, this applies to the components LSP, SPC, SAC, and DTBSV. In practice, the security of these components depends on the underlying platform's capabilities to protect local software. On both target platforms considered in this paper, i.e. Android and iOS, third-party software must be implemented by means of mobile apps. Both platforms feature various security mechanisms that improve the security of installed apps. Examples are sandboxing mechanisms, which isolate installed apps from each other. A detailed overview of security mechanisms integrated into Android and iOS has been provided by (Rogers and Goadrich, 2012) and (Zefferer et al., 2013). In principle, these mechanisms work reliably in practice and provide a sufficient level of security. However, they become useless, if attackers gain root access to the mobile operating system, e.g. by exploiting known vulnerabilities. In this case, the security of installed apps cannot be taken for granted any longer. It can hence be concluded that even though featured security mechanisms provide a certain level of security, absolute security of local components must not be assumed. This applies to both target platforms and implies that locally processed and stored assets are potentially prone to attacks.

While root access is a problem for all local components, several specific aspects need to be considered for the component SAC. The functionality of this component is rather simple and basically limited to obtaining AD from the Signatory and forwarding obtained AD to the QSCD. Because of its limited functionality, the SAC can also be implemented by other means than mobile apps. For instance, in case a local Subscriber Identity Module (SIM) is used as QSCD, a simple SIM application can be used to implement the functionality of the SAC. In this case, the provided security level of SIM applications is relevant. Unfortunately, an attacker with unlimited root access to the operating system must be assumed to have the opportunity to compromise SIM applications as well. Hence, also alternative implementations of the SAC do not provide absolute security. Another relevant aspect is the fact that the SAC potentially needs to implement enhanced authentication functionality in case a remote QSCD is used. In this case, the QSCD does not implicitly implement the authentication factor Possession, as it is not in physical possession of the Signatory. Hence, more sophisticated authentication mechanisms potentially need to be implemented by the SAC, in order to implement multi-factor authentication schemes. Depending on the concrete implementation of these schemes, this can enable additional attack vectors.

Except for the SAC, the achievable security is comparable for all local components that store or process assets. In contrast, the situation is more complex for the different communication-path classes, over which assets are transmitted. The security of assets transmitted between different local software components (Local SW ↔ Local SW) mainly depends on inter-process communication (IPC) capabilities of the underlying mobile platform. The two target platforms Android and iOS differ significantly in this aspect. Android provides broad support for IPC and enables an easy exchange of data between local components. This improves the feasibility of mobile applications, but reduces security, as IPC features can also be employed by e.g. malware to compromise assets. This is especially the case if provided features are used in a wrong way (Chin et al., 2011). On Android, the security of data exchanged between local software components hence depends heavily on a correct use of provided IPC features. On iOS, the situation is less critical, as limited IPC features are provided only. This decreases feasibility, but at the same time reduces the probability of security-critical implementation errors.

While the communication between local software

components is potentially problematic, data exchange between local and remote software (Local SW ↔ Remote SW) can be protected by means of established protocols. For instance, Transport Layer Security (TLS) (Network Working Group, 2008) can be used to reliably assure the confidentiality and integrity of data exchanged between local and remote software. In contrast, the communication path between the Signatory and local software is more problematic. Neither Android nor iOS provide secure input/output capabilities. ARM TrustZone is a potential future solution to this problem, but yet not available on most mobile devices.

Regarding the communication between local software and a local QSCD (Local SW ↔ Local QSCD), a trade-off between feasibility and security can be identified. Android provides mobile apps the opportunity to access local hardware elements that implement QSCD functionality. However, this implies that also malware residing on the mobile device can do so. In contrast, iOS is more restrictive. This reduces the feasibility of ACs relying on local QSCDs on this platform, but also prevents access by malware. Where available, secure communication between local software and local QSCDs must not be taken for granted.

In case of local software and a remote QSCD (Local SW ↔ Remote QSCD), the situation is different. As this setup requires cross-domain communication, additional remote software is needed to extent the QSCD's functionality by means of cross-domain communication capabilities. Again, communication between this module and local software can be easily secured with the help of established protocols such as TLS. The situation is more difficult in case of remote software and a local QSCD (Remote SW ↔ Local QSCD). Again, cross-domain communication is required, which implies the need for an additional module to enhance the QSCD with cross-domain communication capabilities. If this module is implemented as mobile app, communication between this app and the local QSCD is required. This corresponds to the communication-path class Local SW ↔ Local QSCD. If the local SIM is used as QSCD, the required additional module can also be implemented as SIM application. However, this requires a cooperation of the mobile network operator, as communication between remote software and the local SIM needs to rely on the mobile network.

Focusing on the implementation perspective with special regard to the two target platforms Android and iOS shows that security cannot be assured for various components and communication-path classes. By mapping obtained findings to the four ACs, the best AC, i.e. the one that includes the fewest problematic components, can be identified. For a comparative analysis, it is sufficient to focus on those components and communication-path classes, for which there are differences between the four ACs. Figure 3 shows that this applies to one component and four communication-path classes.

Concretely, the SPC is the only component that shows AC-specific differences regarding the three relevant assets. Concretely, the assets SD and DTBS are prone to attacks on the component SPC for AC A and AC B only. Hence, AC C and AC D are advantageous in this regard. Similar conclusions can be drawn for the communication between local software components. This is required for the exchange of assets by AC A and AC B only. As IPC is potentially insecure on the assessed target platforms, AC A and AC B must be regarded as disadvantageous.

The remaining three communication-path classes, for which there are differences between the four ACs, are all related to data exchange between software components and the QSCD. Simply counting the occurrences of assets on these communication-path classes indicates that AC D is advantageous from a conceptual perspective. Following this approach, only the asset AD is transmitted once. For all other ACs, all three assets are present at least once on one of the three communication-path classes. The advantage of AC D is also revealed when taking into account a concrete implementation on the two target platforms. As discussed above, secure communication is easier to achieve for remote QSCDs than for local QSCDs. Thus, AC B and AC D are advantageous in this regard.

4.3 Findings

From the conducted security assessment of the four ACs, several findings can be derived. A comparison of the four ACs on a pure conceptual level yields AC D to be advantageous, as this AC shows the lowest number of components and communication paths, on which identified assets are potentially prone to attacks. Similar findings are also obtained, when the implementation perspective and the current state of the art of mobile platforms is taken into account. Concretely, the conducted assessment has revealed that a local implementation of the SPC is disadvantageous. Furthermore, a local implementation of the QSCD has turned out to be disadvantageous as well. Thus, AC D is also advantageous from an implementation perspective. The other three ACs suffer from the inability to provide a sufficient level of security on current mobile platforms. Although the concrete threat potential depends on the functionality being implemented lo-

cally, AC D is advantageous in any case, as it implements as many components remotely as possible.

5 USABILITY ASSESSMENT

In addition to security, usability has also been identified as crucial success factor. In this section, we hence assess the usability of the four identified ACs. More precisely, we assess the ACs' capabilities to serve as basis for concrete usable implementations. For this purpose, the followed methodology is introduced first. Based on this methodology, the usability assessment is then conducted. Finally, relevant findings obtained from the conducted assessment are summarized.

5.1 Methodology

As both chosen target platforms provide similar input and output capabilities, the basic level of usability provided by Android and iOS is the same. The usability of a mobile signature solution hence only depends on its underlying architecture and implementation. In contrast to the conducted security assessment, platform specifics therefore do not need to be taken into account. Instead, the usability of the four ACs can be compared on a completely platform-independent level. Accordingly, the following methodology has been followed for the conducted usability assessment. First, usability-influencing aspects are derived from related scientific work. The four ACs are then assessed and ranked according to the derived aspects. This finally yields the AC that is best suited to serve as basis for usable signature solutions.

5.2 Assessment

Following the defined methodology, the four ACs are assessed in this section. For this purpose, relevant usability aspects are identified first. The assessment is then based on these aspects.

5.2.1 Usability Aspects

Following the defined methodology, relevant usability aspects are derived from related scientific work. Unfortunately, there is hardly any specific work on relevant usability factors for mobile signature solutions. The probably best suited approach for the present use case has been introduced by (Harrison et al., 2013), who have proposed the PCMAD usability model for mobile applications. This model defines the aspects Effectiveness, Efficiency, Satisfaction, Learnability, Memorability, Errors, and Cognitive Load to be crucial for mobile applications. Recent usability analyses of signature solutions for classical end-user devices have shown that the need for certain software and hardware can also be a usability-reducing factor (Zefferer and Krnjic, 2012). By combining this finding with the PCMAD model proposed by (Harrison et al., 2013), the following set of relevant usability aspects can be derived: Effectiveness, Efficiency, Satisfaction, Hardware Independence, Software Independence, Learnability, Memorability, Cognitive Load, and Error Robustness. Based on these aspects, the usability of the four ACs is analyzed in the following subsection.

5.2.2 Usability Analysis

The usability assessment of the four ACs is based on nine usability aspects. In the following, each of these aspects is discussed separately for the four ACs. In addition, the four ACs are ranked according to their capability to comply with the respective aspect.

(Harrison et al., 2013) describe the aspect Effectiveness as the *'ability of a user to complete a task in a specified context'*. In general, all ACs provide service availability, i.e. allow the successful completion of signature-creation processes. However, service availability is reduced, if required remote components cannot be accessed, e.g. because of a lacking Internet connection. In this regard, AC A is advantageous, as it implements locally as many components as possible. Service availability can also be reduced by the need for additional entities. Solutions based on AC C typically require the Signatory's mobile network operator to access the local QSCD. In roaming scenarios, solutions based on AC C are hence problematic. In summary, AC A is ranked best, as it totally avoids remote components and is independent from additional entities. AC C is ranked last, as it relies on remote components and typically requires on an additional external entity.

For the usability aspect Efficiency, the situation is slightly different. Efficiency defines the speed and the accuracy, with which an intended task can be completed by the user (Harrison et al., 2013). With regard to mobile signature solutions, efficiency is mainly affected by the complexity of the required user-authentication process, as this is the only mandatory user interaction. In general, user authentication is typically more complex in case of remotely implemented QSCDs. In this case, the QSCD is not under physical control of the Signatory and hence cannot implicitly cover the authentication factor possession. Therefore, more complex authentication schemes must be applied to assure an adequate level

of security, which increases complexity. Accordingly, AC A and AC C are by trend more efficient than AC B and AC D.

The factor Satisfaction describes *'the perceived level of comfort and pleasantness afforded to the user through the use of software'* (Harrison et al., 2013). The aspect satisfaction is hence mainly influenced by the provided user interface (UI). As the UI rather depends on the concrete implementation than on the underlying architecture, a conceptual comparison of the four ACs is difficult. Nevertheless, AC C can be identified as slightly disadvantageous. Its reliance on a local QSCD and a remote SPC potentially requires reliance on alternative technologies such as SIM applications. These technologies provide fewer opportunities to implement satisfactory UIs. Accordingly, AC C is ranked worse compared to the other three ACs with regard to the aspect Satisfaction.

The usability aspect hardware independence must be considered, as the need to acquire and maintain additional hardware potentially reduces usability. This has been shown by means of several usability tests. With regard to the four ACs, AC A and AC C must be regarded as disadvantageous. Accordingly, they are ranked worse than AC B and AC D. Similar to hardware independence, also software independence is a crucial usability aspect. Again, the need to acquire and maintain additional software potentially reduces usability. AC A and AC B implement the SPC locally and hence realize more software components on the mobile end-user device than AC C and AC D. Hence, with regard to the aspect Software Independence, AC C and AC D are advantageous and are ranked better than AC A and AC B.

The ability of a user to learn how to use an application is covered by the aspect Learnability and also contributes to an application's overall usability. The learnability of an application is closely related to its complexity, which is measured by the aspect Efficiency. Hence, similar conclusions can be drawn for the aspect Learnability as for the aspect Efficiency. ACs requiring a more complex user-authentication scheme are more complex and hence more difficult to learn. Accordingly, AC B and AC D are ranked worse than AC A and AC C. Similar results can also be obtained for the aspects Memorability and Cognitive Load. Memorability describes the ability of a user to retain how to use an application. Cognitive load describes the impact that using the respective application has on the performance of other tasks that are carried out in parallel. Similar to learnability, memorability and cognitive load are both related to the application's complexity. Hence, also for these aspects, AC A and AC C are advantageous.

Finally, error robustness has been identified as crucial usability aspect as well. For a conceptual comparison of different ACs, it can be assumed that remotely implemented components are less prone to errors. This is reasonable, as remote components can be implemented in data-processing centers that feature redundancy mechanisms to assure required service levels. Under this assumption, the error robustness of an AC increases with the number of remote components. This yields AC D to be the most advantageous solution followed by AC B and AC C. AC A shows the poorest error robustness.

5.3 Findings

Figure 4 summarizes the results of the conducted usability assessment. For each identified aspect, Figure 4 shows the determined ranking of the four ACs. In addition, an overall ranking is derived from the aspect-specific results. This yields AC A as the most usable AC. AC C and AC D share the second rank.

While results obtained from the conducted usability assessment seem quite clear, they suffer from several limitations. First, usability has been assessed by means of abstract architectures only. In practice, usability is also heavily influenced by the concrete implementation of a specific architecture. Second, derivation of the overall ranking has implicitly assumed that all usability aspects are equally important. This is usually not the case in practice, where the relevance of different aspects depends on the concrete context.

Nevertheless, the conducted usability assessment yields several useful findings, as it clearly indicates, which architectures are beneficial regarding which usability aspects. Obtained results also show that AC D is the most bipolar AC. For most assessed aspects, AC D is either among the best or among the worst candidates. Most usability aspects, for which AC D is disadvantageous, are related to the aspect efficiency. Hence, the main usability drawback of AC D is apparently its reduced efficiency, which is mainly caused by the need for more complex user-authentication schemes. If this drawback can be removed, AC D will represent the most usable AC.

6 CONCLUSIONS

The continuing success and popularity of mobile computing raises the requirement for signature solutions that can be applied and used on mobile end-user devices. In this paper, this problem has been addressed by identifying possible architectures for such

	Architecture Candidates (ACs)			
	AC A	AC B	AC C	AC D
Effectiveness	1	2	4	2
Efficiency	1	3	1	3
Satisfaction	1	1	4	1
Hardware Independence	3	1	3	1
Software Independence	3	3	1	1
Learnability	1	3	1	3
Memorability	1	3	1	3
Cognitive Load	1	3	1	3
Error Robustness	4	2	2	1
Sum	16	21	18	18
Overall Ranking	***1***	***4***	***2***	***2***

Figure 4: Usability assessment.

solutions. In total, four ACs denoted as AC A, AC B, AC C and AC D have been systematically derived that basically cover all possible implementation variants. Pros and cons of the four ACs have been revealed by means of a security and usability assessment. With regard to security, AC D has turned out to be advantageous, as it avoids to a large extent the implementation of components on potentially insecure mobile devices. With regard to usability, AC A has been determined as best alternative. AC D has reached the second rank and has also been identified as most bipolar alternative, whose usability could be significantly improved by means of efficient user-authentication schemes. Overall, AC D can hence be identified as the overall winner, if concrete solutions basing on this architecture manage to implement appropriately efficient user-authentication schemes.

All models and architectures proposed and developed in this paper have been intentionally kept on an abstract level. The same holds true for conducted assessments, which have also mainly been applied on a conceptual level. Staying on an abstract level might appear disadvantageous at a first glance, as this approach does not immediately yield a concrete implementation or solution. However, the mobile market is currently undergoing fast and frequent technological changes. Proposing and developing a concrete solution that bases on the current state of the art is hence not sustainable. Therefore, the solution developed in this paper remains independent from the current state of the art. This way, this paper provides a sustainable architectural basis for future mobile signature solutions and paves the way for transactional e-services on mobile end-user devices in the long term.

REFERENCES

A-Trust (2015). Handy-Signatur - Your digital identity. https://www.handy-signatur.at.

Agência para a Modernização Administrativa (2015). Cartão de Cidadão. http://www.cartaodecidadao.pt.

Al-Hadidi, A. and Rezgui, Y. (2009). Critical Success Factors for the Adoption and Diffusion of m-Government Services: A Literature Review. In *Proceedings of the European Conference on e-Government, ECEG*, pages 21–28.

Al-khamayseh, S., Lawrence, E., and Zmijewska, A. (2007). Towards Understanding Success Factors in Interactive Mobile Government.

ANSI (2005). Public Key Cryptography for the Financial Services Industry, The Elliptic Curve Digital Signature Algorithm (ECDSA). http:// webstore. ansi.org/ RecordDetail.aspx?sku=ANSI+X9.62%3A2005.

Apple (2015). iOS 8. https://www.apple.com/at/ios/.

CEN (2004). CWA 14169 - Secure Signature-Creation Devices "EAL 4+". Technical report, European Committee for Standardization.

Chin, E., Felt, A. P., Greenwood, K., and Wagner, D. (2011). Analyzing Inter-Application Communication in Android. In *Proceedings of the 9th International Conference on Mobile Systems, Applications, and Services, MobiSys 2011*, MobiSys '11, pages 239–252. ACM Press.

Common Criteria (2013). Common Criteria. http://www.commoncriteriaportal.org/.

El-Kiki, T. (2007). mGovernment: A Reality Check. In *Conference Proceedings - 6th International Conference on the Management of Mobile Business, ICMB 2007*, page 37. IEEE.

El-Kiki, T. and Lawrence, E. (2006). Mobile User Satisfaction and Usage Analysis Model of mGovernment Services. In *Proceedings of the Second European Mobile Government Conference*, pages 91–102.

Enck, W., Ongtang, M., and McDaniel, P. (2009). Understanding Android Security. *IEEE Security & Privacy*, 7:50–57.

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

Google (2015). Android. https://www.android.com/.

Harrison, R., Flood, D., and Duce, D. (2013). Usability of Mobile Applications: Literature Review and Ratio-

nale for a New Usability Model. *Journal of Interaction Science*, 1(1):1.

ID.ee (2015). Mobiil-ID. http://id.ee/index.php?id=36881.

Karan, K. and Khoo, M. (2008). Mobile Diffusion and Development: Issues and Challenges of m-Government with India in Perspective. In *Proceedings of the 1st International Conference on M4D Mobile Communication Technology for Development*, pages 138–149.

Leitold, H., Hollosi, A., and Posch, R. (2002). Security Architecture of the Austrian Citizen Card Concept. In *18th Annual Computer Security Applications Conference, 2002. Proceedings.*, pages 391–400.

mobiForge (2015). Mobile software statistics 2014. http://mobiforge.com/research-analysis/mobile-software-statistics-2014.

Network Working Group (2008). The Transport Layer Security (TLS) Protocol Version 1.2. http://tools.ietf.org/rfcmarkup/5246.

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

Rogers, M. and Goadrich, M. (2012). A Hands-on Comparison of iOS vs. Android. In *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education*, SIGCSE '12, page 663, New York, NY, USA. ACM.

The European Parliament and the Council of the European Union (1999). Directive 1999/93/EC of the European Parliament and of The Council of 13 December 1999 on a Community framework for electronic signatures. http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2000:013:0012:0020:EN:PDF.

The European Parliament and the Council of the European Union (2014). Regulation (EU) No 910/2014 of the European Parliament and of The Councilof 23 July 2014 on electronic identification and trust services for electronic transactions in the internal market and repealing Directive 1999/93/EC. http://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=CELEX:32014R0910&from=EN.

Zefferer, T., Kreuzhuber, S., and Teufl, P. (2013). Assessing the Suitability of Current Smartphone Platforms for Mobile Government. In *Technology-Enabled Innovation for Democracy, Government and Governance*, pages 125–139.

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