Agile Smart-device based Multi-factor Authentication for Modern Identity Management Systems

Thomas Lenz and Vesna Krnjic
Egovernment Innovation Center - Austria, Inffeldgasse 16a, Graz, Austria

Keywords: Multi-factor, Authentication, Distributed Signatures, Identification, Identity Management, Agile, Provable.

Abstract: Identification and authentication are essential processes in various areas of applications. While these processes are widely described and examined in respect to Web applications that are used on personal computers, the situation is more demanding on smart or mobile devices, as these devices provide other interfaces and have a different user behavior. Additionally, the smart or mobile technology sector has a continuous enhancement that results in no stable technology over the years. Consequently, new usable, agile, and secure methods become necessary to bring identification and secure authentication on smart or mobile platforms. Several proposals are published that should solve this open problem. However, all of them lacks in respect to high-secure authentication by using smart or mobile devices only. In this paper, we propose an agile smart-device based multi-factor authentication model to close the open gap on high-secure authentication. This proposed model combines multiple authenticators on client-side only to increase the assurance of authentication on an eID consumer service. We illustrate the practical applicability of our model by implementing all needed components. Finally, we evaluate the implemented components during the first test in a small group and we a are currently for a wider pilot to evaluate the usability of our proposed model.

1 INTRODUCTION

Electronic identity (eID) is indispensable for a variety of Internet services or electronically coupled devices that perform electronic transactions. Such transactions can include social network interactions, but also more security-sensitive services such as a tax declaration or an eHealth application that protects personal medical data. In each case, besides using an electronic identity authentication is additionally required to prove a claimed identity to be authentic. In more detail, this authentication step links the identity information to an entity, which uses an *Authenticator* to prove that he or she is the owner of that identity information. Consequently, the importance for a high level of assurance by secure means of authentication linked to a qualified identity is rising sharply.

One of the first and still common forms of authentication is the simple provision of user name and password. However, as shown in (Florêncio et al., 2014) passwords are not reliable to provide adequate protection for security relevant applications, because latest work on password security (Taneski et al., 2014) shows that users and their passwords are still considered the weakest link in a process-flow for entity authentication. The security and reliability of entity authentication-process can be increased by using more than one authentication factors. This concept is called multi-factor authentication. Multi-factor authentication use two or more authentication factors from different categories to increasing the reliability of the authentication process. More and more Internet services and Web-based applications offer their users authentication by using a multi-factor approach. Multi-factor authentication is mandatory for state-of-the-art implementations of security-sensitive Internet services like eHealth applications or transactional eGovernment services. To maintain a sufficient level of security, implemented authentication processes and involved user devices has to keep pace with the technological developments and must react immediately on changing threat scenarios.

This requirement can become challenging in practice, as many technologies are often not flexible enough to keep pace with evolving requirements. The mobile sector is a very illustrative example of an area that is developing very fast. Due to the fast increasing usage of different devices, a smooth interaction between systems and solutions that interact with these devices, qualified identification and secure authentication get necessary. Especially in the smart or mobile sector, we face the challenge of providing secure,

Lenz, T. and Krnjic, V.
Agile Smart-device based Multi-factor Authentication for Modern Identity Management Systems.
DOI: 10.5220/0007229601130124
In *Proceedings of the 14th International Conference on Web Information Systems and Technologies (WEBIST 2018)*, pages 113-124
ISBN: 978-989-758-324-7

usable, and agile authentication methods, because it is not possible to define an authentication method or in general, a technology that is stable over the years. Consequently, a new usable and agile but also secure authentication method becomes necessary for these platforms.

While identification and secure authentication are widely described and examined, in respect to Web applications and services that are used on personal computers (PC) or similar devices, the situation is more demanding on smart or mobile devices. Typically, smart cards are in use for identification and high-secure authentication that implements two-factor authentication approach. The smart cards are inserted into a card reader connected to the personal computer. However, many smart or mobile devices, like smart phones or tablets, cannot connect to card readers. To overcome this issue, server-based solutions, like the Austrian Mobile-Phone Signature, are evolved. Nevertheless, there are also problems on some devices, because they could not have a sufficient user-interface to handle the identification and authentication process, or they need cellular radio to receive short messages (SMS) for mobile tan (mTAN) to implement authentication by using two authentication factors. Especially about mTAN based solutions, there were many security incidents in the last couple of years (Hayikader et al., 2016; Haupert and Mller, 2016).

To overcome this issue, different solutions are proposed. One solution is to use wireless communication to interact with a contactless smart card. Such approach can be used on smartphones or tablet computers and was already described in (Ferraiolo et al., 2014b). However, sufficient contactless communication interface, like NFC is not available on every device. Such solutions are implementable in a secure way but lacks interoperability or usability. Therefore, another approaches are desirable.

Such approach could be the transfer of entity information, which based on an already existing strong authenticated method, from one device to another device. In such a scenario, a user could authenticate a session on a personal computer by using an existing eID, like smart-card based, that facilitates qualified identification and secure authentication. After this initial identification and authentication step, the identification information can be transferred and bind to another smart or mobile device on which the eID information can be reuse later for identification and authentication. Such transfer of an already authenticated session to a smart or mobile device is similar to the guidelines for derived personal identity verification (PIV) credentials, which was published in the NIST Special Publication 800-157 (Ferraiolo et al., 2014a). There, the NIST describes guidelines and requirements for derived PIV credentials, which are based on the general concept of derived credentials in NIST Special Publication SP 800-63-2 (Burr et al., 2013). Francisco Corella and Karen Lewison (Corella and Lewison, 2012; Corella and Lewison, 2014) and the Entrust Datacard company (Entrust, 2014) already published some examples of a derived credential architecture. However, these already existing solutions are focusing on specific requirements in enterprise ID systems that are often used by companies but describe no general process or architecture. Therefor, there exists no sufficient solution to use already existing qualified eID by using secure authentication methods on smart or mobile devices for identification and authentication on any other service providers or eID consumers in general.

One example to solve these problems was already proposed by (Lenz and Alber, 2017). This solution based on a generic concept of cross-device eID that provides identification and secure authentication to almost all service provider by using smart or mobile devices by using security features that are shipped with current smart or mobile devices. However, smart or mobile devices are different to smartcards regarding security features or security certifications, because most of them are open or semi-open platforms that facilitate the execution of different applications. So, the current security features provided on smart devices, like as Sandboxing[1] for separating of running applications or hardware-based cryptographic elements [2] to manage cryptographic keys provide an obvious higher security level than simple password-based authentication or two-factor authentication by using password and mTAN. However, the missing security certification and the design-related semi-open platform of mobile devices limits the reliability into a mobile device as a single authenticator.

To antagonize this lack of reliability into a single mobile-device based authenticator, we propose an advanced multi-factor based approach for entity authentication that cryptographically combines at least two cryptographic key-pairs that are located on different authenticators on entity side. Therefor, this cryptographic combination can be accomplished on the smart or mobile device itself without influencing already existing eID consumer services. By using this approach, there is not implementation update required on identity-provider side or eID-consumer side. Consequently, our proposed solution fulfills the requirements for modular and flexible identity mana-

[1] https://techterms.com/definition/sandboxing
[2] https://source.android.com/security/keystore/

gement systems described in (Lenz and Zwattendorfer, 2015). Furthermore, we can increase the security into authentication on the client side only by combining different cryptographic keys that are located on different devices or tokens. In other words, it is not possible anymore for an attacker, which compromises the smart or mobile device, to use the device as an authenticator. Consequently, this approach decreases the likelihood of successful attack the authentication process-flow significantly, because it is not enough for an attacker to compromise one single device. Also, this approach has no special requirements into the second authenticator that are different from the requirements defined in (Lenz and Alber, 2017). Therefore, this solution perfectly fits into the already existing concept of cross-device eID and increases the reliability into entity authentication significantly. In a nutshell, we propose an extension to the idea of personal, derived credentials (PIV) by adding multi-factor authentication on client side to the concept of cross-device eID. By using this approach, we can increas the reliability into authentication significantly without influence the eID consumer server. This paper is structured as follows. Section 2 defines some terms, like eID, authentication, authentication factor that are used in this paper. In Section 3, we shortly present the architectural concept of cross-domain eID. After this, we give technical details on our proposed model for multi-factor combination on entity device-side to increase the entity authentication assurance and illustrate the integration into the concept of cross-domain eID. Section 4 gives detail information about the practical implementation of the proposed model. Finally, evaluation-related aspects are detailed in Section 5. Finally in Section 6 we give a short conclusion.

2 DEFINITIONS

The aim of this section is to define some terms regarding eID and authentication to build up a basis for all further concepts discussed in this paper. We start with the definition of electronic identity (eID). After this, we define the term *authentication* and illustrate additional aspects, like multi-factor authentication. Finally, we define the term *Domain* that is multiple used in this paper to build up a link to *Cross Device*.

2.1 eID

A precise definition of electronic identity (eID) or identity in general is hard to give because a verity of definitions exists. Every of this definition has a different meaning according to the semantic context and the applied environment. As an example, a social scientist defines identity as an: *"Identity is an umbrella term used throughout the social sciences to describe an individual's comprehension of him or herself as a discrete, separate entity."*[3] A common aspect of all definitions is that identity means the presentation of an entity in a particular domain. So, an electronic identity is the digital representation of an identity (Sarikhani, 2008; Jøsang et al., 2007; ISO/IEC, 2016; Zwattendorfer, 2014). This reference to a particular domain is also part of the ISO/IEC FDIS 24760-1 specification (27, 2011) for Identity. The digital representation consists of an identifier, attributes which characterize additional properties of the entity, and credentials that provide evidence claims of the digital entity. As an example, the Commission Implementation Regulation (EU) 2015/1501 on the eIDAS interoperability framework (European Union, 2015) published at 8. September 2015 a minimum data set of attributes, that has to be included to an eID. However, the eIDAS interoperability framework defines a minimum set of attributes. The identifier and consequently the digital identity had not been unique and persistent in general, as it could only be valid within a certain time frame or in a specific context. Concisely, the term identity or its electronic representation describes the distinct and non-ambitious properties and characteristics of an entity.

2.2 Authentication

Authentication is the process to provide evidence for a claimed digital identity with a certain level of assurance. In more detail, authentication means a formal technical or organizational process to get evidence that the digital identity, which is shown by an entity, is authentic and that the entity is the owner of the digital identity. The formal process uses one or more authentication factors to get evidence into the identity. If this formal authentication process is successfully finished, it results in an authenticated identity. (Grassi et al., 2017; ISO/IEC, 2013; 27, 2011)

2.3 Authentication Factor

An authentication factor is a piece of information or a part of the authentication process that is used to authenticate or verify the identity of an entity. Many related work and standards (Mohammed and Elsadig, 2013; Kim and Hong, 2011; ISO/IEC, 2013; Grassi et al., 2017) defined different types of authentication

[3] http://ezinearticles.com/

factors. These authentication factors are grouped into three different categories:

- **Something a User Knows:** Secrets such as a password or a PIN.
- **Something a User Is:** Biometric factors such as a fingerprint or an iris recognition.
- **Something a User Has:** Devices such as tokens or smart cards dedicated to an entity.

So, an authentication factor is the technical or organizational implementation of specific authentication sub-process. This implementation is called *Authenticator* and implements at least one authentication factor (Grassi and Feton, 2017).

2.4 Multi-Factor Authentication

Multi-factor authentication is an authentication that uses two or more authentication factors from different categories. Multi-factor authentication can be performed by using a single authenticator that provides more than one factors from different categories or by a combination of different authenticators that provides one factor (Grassi and Feton, 2017).

2.5 Domain

A single precise definition of Domain does not exist because the term is widely used in different fields of science and business. From a term base point of view, the term Domain based on the Latin term *Dominium* that is a legal term meaning *control* or *ownership* [4] [5] [6]. The specification ISO/IEC FDIS 24760-1 (27, 2011) specifies a Domain as an environment where an entity can use a specific set of attributes for identification and other purposes. In respect to this definition from ISO for Domain, the identification means the process of recognizing an entity in a particular domain as distinct from other entities. However, all definitions have in common that the term Domain means an administrative district or a subzone that has its characteristics and requirements. In respect to this paper, a domain is a system or a part of a system that has specific requirements into eID information or into an eID token that provides eID information. Cross-domain in respect to this proposed model means every transition from one system to another system in which the provided behavior and the required behavior does not match.

Cross-device focus on another level of cross-domain interoperability that tackles with the use of eID information and secure authentication between different devices. Regarding the definition of Domain and the use of eID and authentication information on smart or mobile devices, a smart or mobile device can also be a discrete domain because not every existing eID source can be used on each smart or mobile device since e.g. technical requirements are not met. To overcome this issue, cross-device eID bridge the gap to use already existing identity or authentication approaches on different devices. Consequently, cross-device is not completely independent from cross-domain since it is a different angle of view on the usage of eID.

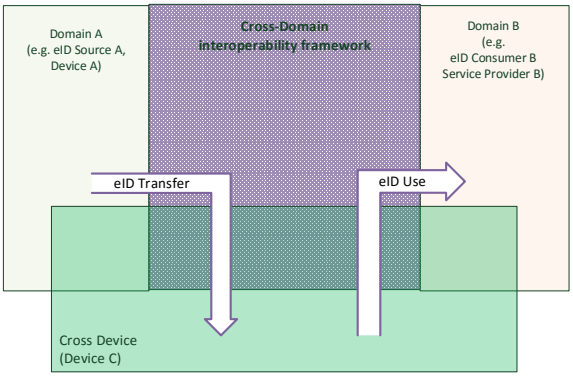

Figure 1: Links between Domain, Cross Domain and Cross Device.

3 MODEL

This section illustrates the concept of cross-domain eID (Lenz and Alber, 2017) and gives detail information about our model to facilitate user-centric eID and secure authentication on semi-trusted smart or mobile devices. In Subsection 3.1, we shortly describe the concept of cross-domain eID and define the stakeholders that are involved in the eID lifecycle in this architecture. Afterward, Section 3.2 gives more detailed information about the authentication process-flow that is used in the concept of cross-domain eID. In Subsection 3.3, we describe our proposed advanced multi-factor combination model for semi-trusted smart devices that facilities high secure authentication and user-centric eID.

3.1 Concept of Cross-Domain eID

Figure 2 shows the concept of cross-domain eID that mainly focus on qualified eID in a privacy preserving and user-centric model. The entities or stakeholders, which are involved into the eID processing, are partially similar to the stakeholders involved in a classic

[4] http://www.thefreedictionary.com/dominium
[5] http://www.dictionary.com/browse/dominium
[6] https://en.wikipedia.org/wiki/Dominium

identity management system (Bertino and Takahashi, 2010), however the interactions between the stakeholders and the assignments regarding trust relationships are different. The following itemization gives a short description of the stakeholders illustrated in Figure 2.

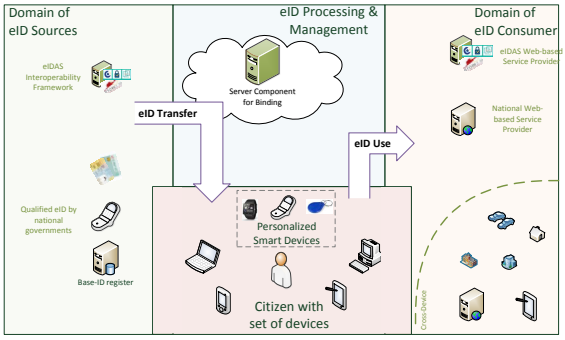

Figure 2: High-level idea of Cross-Domain eID.

- **eID Source:** The stakeholders on the left side represent a generic set of eID sources that provide an electronic identity for an entity in a specific context. In case of qualified eID, the eID source can be a registration authority (RA), like a national register that acts as a trusted third part for eID attributes and issues qualified eID attributes to the entity or other stakeholders.
- **eID Consumer:** The area on the right side represents an application, service, or device that depends on the identification information. Consequently, the eID consumer has to be in confidence into the eID information that was provided by an entity in a user-centric approach. In the concept of cross-domain eID, such eID consumer can be located in different domains and therefore the eID consumer needs eID information for a specific context. For example, there can be a legal requirement that an eID consumer services need different identifiers of an entity concerning the domain of the eID consumer for privacy reasons.
- **Entity:** An entity or user wants to access a protected resource of an eID consumer, like service provider. Therefore, the eID consumer consumes the eID information selective revealed by the entity to grant or deny access. The eID information, which was issued by an eID source, is stored on a personalized smart device in this model.
- **Personalized Smart Devices:** The personal smart device that is illustrated in the bottom area is a subtotal set of devices used by an entity to interact with eID consumer services by using identification information issued from an eID source. In our model, these set of devices are modeled as semi-trusted. There is a basic trust relationship according to the security features and there implementation of the smart or mobile device, like Sandboxing or hardware-based cryptographic elements. However, there is less confidence due to high-secure authentication by using these features on the same level as expressed for example by smart cards.

3.2 Agile Mobile Authentication for Smart and Mobile Devices

In respect to the high-level model illustrated in Figure 2, the proposed process for agile mobile authentication consists of two sub-processes. The first sub-process is the binding process that transfers and binds eID information to a smart or mobile device by using cryptographic operations. The second sub-process is the eID usage process. During the eID usage process, an entity uses a smart or mobile device, which is already bound to an eID, for identification and authentication on an eID consumer service in a secure manner.

Figure 3 illustrates the generic binding process of the agile mobile authentication algorithm. This process cryptographically binds an eID that is derived from an existing entity eID to a smart or mobile device. The process shown in Figure 3 consists of the following steps.

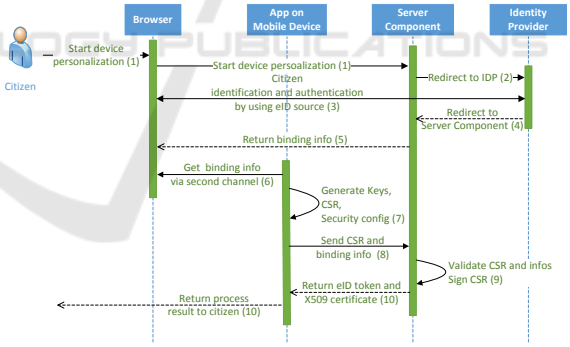

Figure 3: Generic process to create a cryptographic binding for device personalization.

1. An entity wants to personalize its smart or mobile device by using an existing eID. For this purpose, the entity requests the Server Component that provides a binding service to personalize devices.
2. The Server Component redirects the entity to an Identity Provider (IDP) for initial identification and authentication. This identification and authentication use an existing eID that is provided by an eID source.
3. The entity execute the identification and authentication process by using its existing eID. This step

satisfies the first requirement to support of any eID source.

4. If the initial identification and authentication is finished, the IDP sends the existing eID information to the Server Component. After this step, the eID derivation process is almost finished.
5. The Server Component starts the cryptographic binding process. Therefore, the Server Component provides generic binding info that contains all information that is required to initialize the binding part of the agile mobile authentication process. This binding info is provided to the entity and the smart or mobile device by using a second channel that is not fixed to a specific technical solution in general.
6. The entity uses a binding application on its smart or mobile device to receive the initial binding info over the second channel.
7. If the application receives the initial binding info than the cryptographic part of the binding process starts. At first, the application generates a public/private key pair by using security features provided by the smart or mobile device. After this, the application build a certificate-signing request (CSR) (Turner, 2010) by using the pubic key generated before. At last, additional security measures can be set by the entity to restrict the use of the private key on the smart or mobile device. This restriction can be a PIN/password in the simplest case, but also some biometric factors like fingerprint, if it is supported by the smart or mobile device.
8. In the next step, the application connects to the Server Component and sends the CSR and the binding info.
9. The Server Component validates the CSR and the binding info. If both are valid, then the Server Component signs the CSR to generate an X509 certificate. The X509 certificate is attached to the eID information, which was received from the IDP in step four. The Server Component also signs the extended eID information. After this, the eID derivation process is completed.
10. In the last step of the binding process, the derived eID information is sent to the smart or mobile device and can be stored by the application. The result of the binding process is shown to the entity.

If the binding process was successfully finished than the entity can use the smart or mobile device for authentication. Figure 4 illustrates the generic usage process of the agile mobile authentication algorithm.

This generic process flow shows the usage of a derived eID that is cryptographically bound to a smart or mobile device for authentication on a Service Provider. A detailed description of this process is given in the following.

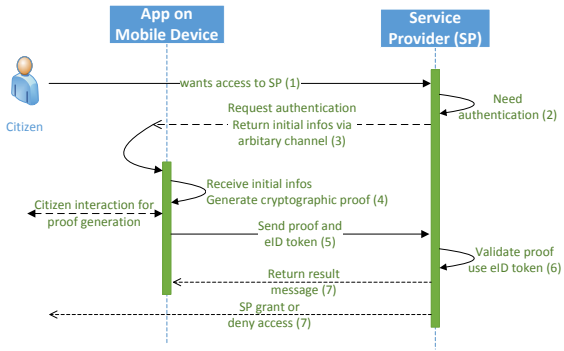

Figure 4: Generic process to use a personalized device for authentication.

1. An entity wants access to a Service Provider. This Service Provider is not fixed to a specific type, as a Web-based application as an example, but could be any service that requires identification and secure authentication.
2. The Service Provider validates the access request and request authentication from the entity if it is needed.
3. To start the proposed agile mobile authentication process, the Service Provider provides all information that is necessary to initialize the algorithm over an arbitrary channel. This arbitrary channel is not fixed to a specific technology to satisfy the requirement to support almost all eID consumer services.
4. The entity can use its smart or mobile device to receive the information from the Service Provider. The application generates a cryptographic proof, by using the private key that was generated in the binding phase. If the entity has restricted the access to the private key in the binding phase, then also additional entity related information is necessary to complete this cryptographic proof.
5. The application sends the cryptographic proof and the derived eID information to the Service Provider.
6. The Service Provider can validate the derived eID information and the proof by using the X509 certificate that is attached to the derived eID information. If the validation is successful than the agile mobile authentication process is finished and the derived eID information can be used to identify the entity.

7. The Service Provider returns the result of the validation to the application, and after this, the entity can access the restricted area on the Service Provider.

3.3 Multi-factor Combination on Semi-trusted Smart and Mobile Devices

In this subsection, we illustrate our proposed model which cryptographically combine at least two authenticators that implement different authentication factors on a semi-trusted smart or mobile device to increase the reliability into the authentication process describe in Section 3.2. In more detail, the proposed model enhances the management of cryptographic keys and increases the trust into cryptographic keys that are created during the cryptographic binding process (see Figure 3, Step 7) or are used during the authentication process (see Figure 4, Step 4). Consequently, our advanced model for multi-factor combination on client side perfectly fits into the existing cross-domain eID approach, because these improvements do not influence other stakeholders besides the entity and the personalized smart device.

From a cryptographic point of view, threshold cryptography is used to cryptographically combine different multiple authentication factors that are implemented as authenticators on different smart or mobile devices. While threshold cryptography itself is no new cryptographic scheme, and a large body of research was done around the problem in most general form (Boyd, 1986; Croft and Harris, 1989; Fiat and Shamir, 1987; Gennaro et al., 2001; MacKenzie and Reiter, 2004; Schnorr, 1990; Lindell, 2017), the interest on threshold cryptography has been renewed for the purpose of key protection or distributed signatures schemes on semi-trusted devices such as mobile phones or any other smart device. For example such key protection approaches by using threshold cryptography can be used in bitcoin to protect the private signing key. However, our proposed model uses threshold cryptography to distribute the signature generation capabilities to different authenticators, which means that more than on cryptographic key is needed to generate a valid signature.

Threshold cryptography schemes for distributed signature generation exists for a wide variety of digital-signature schemes like RSA signing, digital signature schemes (DSA) based on RSA or elliptic curves (ECDSA), or other signature schemes like Schnorr signatures (Schnorr, 1990). While it is more complex to build a distributed signature scheme on ECDSA signatures as it is more difficult to find a scheme to compute k and k^{-1} without knowing the private key k, it is much easier to define a distributed signature for other signature schemes. Schnorr signatures based on elliptic curves are one well example for such a signature scheme that facilitate distributed signature generation without complex and time expensive cryptographic operations. Consequently, the elliptic curve version of Schnorr signature schemes is used to integrate distributed signature schemes in our proposed model of cross-domain eID, as the signature generation can be easily integrated into lightweight smart or mobile devices.

According to the concept of cross-domain eID, our proposed model for multi-factor combination on semi-trusted mobile or smart devices can be split into three phases. The first phase is the distributed key generation, which can be integrated into Step 7 of Figure 3, that generates a virtual asymmetric public/private key-pair $key(PK^{\text{binding}}, sk^{\text{binding}})$ used for the cryptographic binding described in Step 7. We called this key pair virtual, because the private key k^{binding} does not exist on one single device, as it is generated dynamically from more than one device specific private keys sk^{device_i}, $i \in (1,...,n)$ where n is the number of devices. In more detail, this generation of the public part of the virtual binding key PK^{binding} can be formal described as $PK^{\text{binding}} = (\sum_{i=1}^{n} sk^{\text{device}_i}) \cdot G$, where G is the generator of the group. Figure 5 illustrates a virtual binding-key generation by using two smart devices.

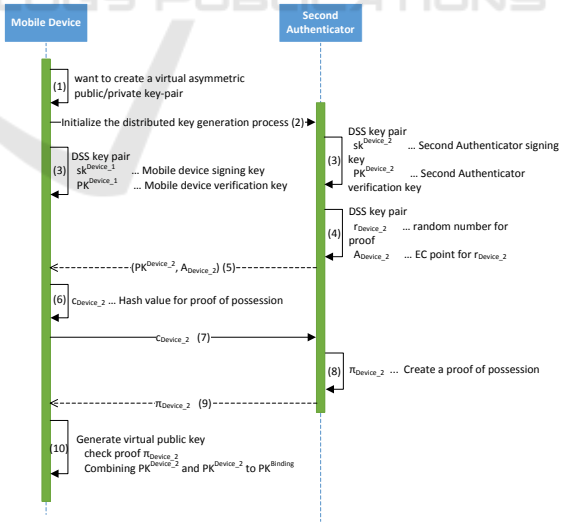

Figure 5: Generation of a virtual public/private key-pair by using two devices.

In the following, we describe this key generation process in a generic form for more than two devices.

1. The smart or mobile device that should be personalized by using an already existing eID and the

authentication should be done by the virtual binding key $key(PK^{\text{binding}}, sk^{\text{binding}})$

2. The personalization device sends a request to every authenticator to start the key generation process.
3. Every smart or mobile device that should be used as an authenticator generates its own asymmetric private key-pair $key(PK^{\text{device}_i}, sk^{\text{device}_i})$. The elliptic curve point that represents the public key PK^{device_i} is generates as $PK^{\text{device}_i} = sk^{\text{device}_i} \cdot G$, where $G \in E/F$ is the generator. This device specific asymmetric key-pair can be located in a hardware-based cryptographic element that is available on the authenticator.
4. Every authenticator generates a second random number $r_i \in F_q$ and a corresponding elliptic curve point $A_i = r_i \cdot G$, where $G \in E/F_q$ the generator is. The point A is required for the authenticator to knows the private key sk^{device_i}.
5. The authenticator sends the set $(PK^{\text{device}_i}, A_i)$ to the smart or mobile device that should be personalized.
6. The smart device calculates the hash value c_i by using a cryptographic hash function H from input data $c_i = H(G, PK^{\text{device}_i}, A_i)$.
7. The smart device sends the hash value c_i to the authenticator i to get a proof of possession of the private key sk^{device_i}.
8. By using c_i, every authenticator calculates a proof $\pi_i = r_i + c_i \cdot sk^{\text{device}_i} \mod q$, where q is modulo of the field F_q.
9. Every authenticator send its prove π_i back to the smart or mobile device that should be personalized.
10. The personalization device checks every proof π_{P_i}. If it is valid, the personalization device adds the authenticator public key PK^{device_i} to PK^{binding}. This simple elliptic curve addition operation can be done because: $PK^{\text{binding}} = (\sum_{i=1}^{n} sk^{\text{device}_i}) \cdot G = \sum_{i=1}^{n} (sk^{\text{device}_i} \cdot G) = \sum_{i=1}^{n} PK^{\text{device}_i}$ If all public keys are added the virtual public binding key PK^{device_i} can be used for the binding process.

The second phase is the distributed signature generation, which can be integrated into Step 4 of Figure 4 that uses the virtual binding key $key(PK^{\text{binding}}, sk^{\text{binding}})$ to sign a challenge which is equivalent to a cryptographic proof. Additionally, this second phase is also required in Step 7 in Figure 3 to sign the CSR in the binding process. Figure 6 illustrates this distributed signature generation process by using two devices.

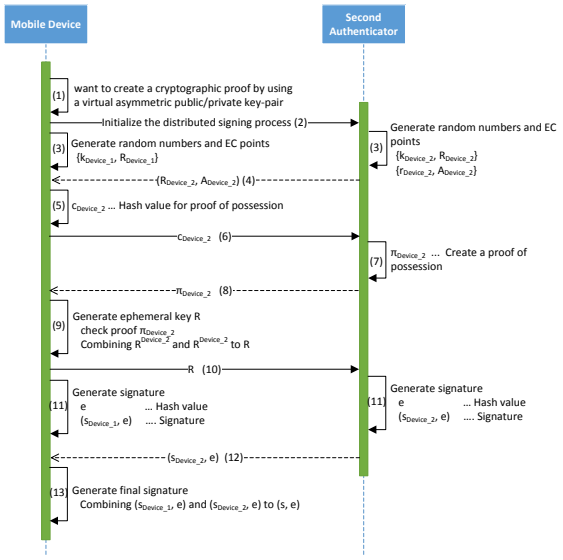

Figure 6: Distributed signature creation by using two devices.

In a generic form and with more detail, a distributed signature generation consists of the following steps:

1. The personalized smart or mobile device is requested by an eID consumer server to generate a cryptographic proof, by using the virtual private key sk^{binding}) that was generated in the first phase. This cryptographic proof can be a digital signature which uses an elliptic curve Schnorr signature scheme on an input message m that was sent by the eID consumer service.
2. The personalized smart or mobile device generates a distributed signature initialization requests.
3. Every authenticator receives the signature initialization request and generates new random $k_i \in F_q$ where F_q is the field over the elliptic curve E. By using k_i, the authenticator can calculates a new random point $R_i = k_i \cdot G$, where $R_i \in E/F$ and G is the generator.

   In addition, every authenticator generates a second random number $r_i \in F_q$ and a corresponding elliptic curve point $A_i = r_i \cdot G$, where $G \in E/F_q$ is the generator. The point A is one step of the proof that the authenticator knows the random value k_i.
4. Every authenticator send the set (R_i, A_i) to the personalized smart device.
5. The smart device calculates the hash value c_i by using a cryptographic hash function H from input data $c_i = H(G, R_i, A_i)$.
6. The smart device sends the hash value c_i to the authenticator i to get a proof of possession of the

private key k_i.

7. By using c_i, every authenticator calculates a proof $\pi_i = r_i + c_i \cdot k_i \mod q$, where q is modulo of the field F_q.
8. Every authenticator sends its prove π_i back to the smart or mobile device that should be personalized.
9. The personalized smart device checks every proof π_i. If it is valid, the personalization device adds the random points R_i to R. This is a simple elliptic curve addition operation as: $R = (\sum_{i=1}^{n} k_i) \cdot G = \sum_{i=1}^{n} (k_i \cdot G) = \sum_{i=1}^{n} R_i$.
10. The personalized smart device generates a distributed signature creation request that contains the message m and the sum of the randomly generates points R.
11. Next, every authenticator generates a cryptographic hash e from input message m and the random point R by calculating $e = H(m||R)$, where H is the cryptographic hash function. At last, the authenticator computes the signature value s by calculating $s = k - sk^{\text{device}_i} \cdot e \mod q$, where k is the random number, e hash value, and sk^{device_i} is the private key of a specific authenticator.
12. Every authenticator sends the signature $\sigma_i(s_i, e)$ back to the personalized device.
13. If all signatures $\sigma_i(s_i, e)$ are received, the personalized smart device can aggregate the single signature $\sigma_i(s_i, e)$, by simple adding the single signature values s_i to $s = \sum_{i=1}^{n} s_i \mod q$. This sample add operation is possible, because s_i was created only by linear operations. After this, the process of signature creation is completed and the signature $\sigma_m(s, e)$ for message m can be used as a proof of possession for the virtual secret key sk^{binding}).

The third phase is the signature verification phase in which an eID consumer service can verify the cryptographic signature that is used to authenticate the entity. This signature verification step is part of Step 6 in Figure 4. From an eID consumer point of view, this verification phase is equal to the verification of an elliptic curve Schnorr signature by using the virtual public key PK^{binding} from the personalized smart or mobile device. More details on cryptographic building blocks for the Schnorr signature scheme and other basic cryptographic primitives that we used in our proposed model for multi-factor combination on mobile or smart devices are described in the Appendix 6l.

We have evaluated the practical applicability of the proposed model for multi-factor combination on mobile or smart devices by realizing two applications for smart devices that implement our proposed model.

4 IMPLEMENTATION OF MULTI-FACTOR COMBINATION ON CLIENT-SIDE

We used our proposed model of client-side multi-factor combination for entity authentication to implement and demonstrate the practical applicability in practice. This implemented solution consists of two applications for smart devices. The first application is a mobile-phone application that stores and manages the eID information in a secure way and implements the first authenticator regarding our proposed model. This mobile phone application can be used as personalized smart device regarding the concept of cross-domain eID. We implement a mobile-phone applications for the Android Operation System (Android OS) [7] that provides all functionality for binding and usage of eID information, by implementing our multi-factor combination approach.

The second one is a smart-watch application that implements a second authenticator for our proposed model. For the second authenticator different smart watches were used, for example, a SmartWatch 3 from Sony[8], but every device runs on the Android Operation System. We implement an application for smart watches that perform all cryptographic operations, which are required in our model. Also, this application has a simple user interface to protect the secret key by using a PIN approach.

From a cryptographic point of view, we use the elliptic curve P-256 from (Kerry et al., 2013) to implement our proposed model of a multi-factor combination. The basic implementation of the Schnorr signature scheme, which is used in our model, was done according to the recommendations from *BSI TR-03111* (BSI, 2018). However, we modify the signature creation described in *BSI TR-03111* according to Section 3.3 to facilitate distributed signature creation. To implement the proof of possession, which is used in Section 3.3, we use the Schnorr NIZK Proof over Elliptic Curve from RFC 8235 (Hao, 2017). By using these cryptographic schemes, we could illustrate the practical applicability of our proposed model by developing two authenticators that are implemented as smart applications.

[7] https://www.android.com/

[8] https://www.sonymobile.com/global-en/products/smart-products/smartwatch-3-swr50

5 EVALUATION

The successful implementation of two authenticator prototypes for multi-factor combination on client-side has shown the feasibility of the proposed model. In order to evaluate the capabilities of our solution in a real-world scenario, we have deployed and tested the implementation with test deployments of real eGovernment infrastructure components. We deployed the server component of the already existing cross-domain eID infrastructure that was used during some evaluation phases in 2016 and 2017 (Lenz and Alber, 2017). By using this deployment, Austrian citizen can use there national eID cards to personalized there mobile phones. To evaluate the use of proposed model for multi-factor combination, we have deployed some demo service-provider that can be used by entities to test the advanced authentication process, which is illustrated in this paper. First internal tests shows practical applicability of our proposed model and illustrates the smooth integration into the existing cross-domain eID infrastructure. Currently, we are in the starting phase of a pilot to evaluate the proposed model in a bigger group of entities to get more detailed information on usability aspects regarding our model.

6 CONCLUSION

In this paper, we have presented an agile smart-device based multi-factor authentication process to facilitate identification and high-secure authentication on any service provider by using smart or mobile devices. Due to the increasing number of smart devices that process sensitive data, identification and authentication on and from smart or mobile devices become indispensable. While eID processes are widely examined for Web-based applications and PCs, the situation is more demanding on smart or mobile devices. A first already published approach brings agile authentication to be a smart or mobile device. However, this approach lacks in respect to high secure authentication, as authentication was done from a single semi-trusted device. To antagonize this lack of reliability into entity authentication, we propose an advanced multi-factor based approach for entity authentication that cryptographically combines at least two key-pairs on entity side. By using this, we can significantly increase the reliability of the authentication process. We have demonstrated the practical applicability of our proposed multi-factor authentication process by implementing all components of the proposed model. A first evaluation pilot is starting to evaluate the usability of our proposed model.

REFERENCES

(2018). Bsi tr-03111: Elliptic curve cryptography, version 2.1.

27, I. J. S. (2011). Information technology Security techniques A framework for identity management Part 1: Terminology and concepts. Technical Report 24760-1, ISO/IEC.

Bertino, E. and Takahashi, K. (2010). *Identity Management: Concepts, Technologies, and Systems*. Artech House, Inc., Norwood, MA, USA.

Boneh, D. (2005). *Schnorr Digital Signature Scheme*, pages 541–542. Springer US, Boston, MA.

Boyd, C. (1986). Digital multisignatures. In *In Cryptography and Coding*, pages 241 – 246.

Burr, W. E., Dodson, D. F., Newton, E. M., Perlner, R. A., Polk, W. T., Gupta, S., and Nabbus, E. A. (2013). Electronic authentication guideline. Technical Report 800-63-2, National Institute of Standards and Technology (NIST).

Chatzigiannakis, I., Pyrgelis, A., Spirakis, P., and Stamatiou, Y. (2011). Elliptic curve based zero knowledge proofs and their applicability on resource constrained devices.

Corella, F. and Lewison, K. (2012). Techniques for implementing derived credentials. Technical report, Pomcor Research in Mobile and Web Technology.

Corella, F. and Lewison, K. (2014). An example of a derived credentials architecture. Technical report, Pomcor Research in Mobile and Web Technology.

Croft, R. A. and Harris, S. P. (1989). Public-key cryptography and reusable shared secrets. In *In Cryptography and Coding*, pages 189 – 201.

Entrust, e. a. (2014). Mobile derived piv/cac credential - a complete solution for nist 800-157. Technical report, Entrust Datacard.

European Union (2015). Commission implementing regulation (eu) 2015/1501 of 8 september 2015 on the interoperability framework pursuant to article 12(8) of regulation (eu) no 910/2014 of the european parliament and of the council on electronic identification and trust services for electronic transactions in the internal market. European Union.

Ferraiolo, H., Cooper, D., Francomacaro, S., Regenscheid, A., Mohler, J., Gupta, S., and Burr, W. (2014a). Guidelines for derived personal identity verification (piv) credentials. Technical Report 800-157, National Institute of Standards and Technology (NIST).

Ferraiolo, H., Regenscheid, A., Cooper, D., and Francomacaro, S. (2014b). Mobile, piv, and authentication. Technical Report Draft NISTIR 7981, National Institute of Standards and Technology (NIST).

Fiat, A. and Shamir, A. (1987). How to prove yourself: Practical solutions to identification and signature problems. In Odlyzko, A. M., editor, *Advances in Cryptology — CRYPTO' 86*, pages 186–194, Berlin, Heidelberg. Springer Berlin Heidelberg.

Florêncio, D., Herley, C., and Van Oorschot, P. C. (2014). An administrator's guide to internet password research. In *Proceedings of the 28th USENIX Conference on Large Installation System Administration*, LISA'14, pages 35–52, Berkeley, CA, USA. USENIX Association.

Gennaro, R., Jarecki, S., Krawczyk, H., and Rabin, T. (2001). Robust threshold dss signatures. *Information and Computation*, 164(1):54 – 84.

Grassi, Paul A. Garcia, M. E. and Feton, J. L. (2017). Digital identity guidelines. Technical Report 800-63-3, National Institute of Standards and Technology (NIST).

Grassi, P. A., Fenton, J. L., Newton, E. M., Perlner, R. A., Regenscheid, A. R., Burr, W. E., and Picher, J. P. (2017). Digital identity guidelines - authentication and lifecycle management. Technical Report 800-63b, National Institute of Standards and Technology (NIST).

Hao, F. (2017). Schnorr Non-interactive Zero-Knowledge Proof. RFC 8235.

Haupert, V. and Mller, T. (2016). (in)security of app-based tan methods in online banking. University of Erlangen-Nuremberg, Germany.

Hayikader, S., Hanis binti Abd Hadi, F. N., and Ibrahim, J. (2016). Issues and security measures of mobile banking apps. *International Journal of Scientific and Research Publications*, 6.

ISO/IEC (2013). ISO/IEC 29115. Information technology – Security techniques – Entity authentication assurance framework. International standard, International Organization for Standardization.

ISO/IEC (2016). ISO/IEC COMMITTEE DRAFT 29003. Information technology – Security techniques Identity proofing. Technical report, International Organization for Standardization.

Jøsang, A., Zomai, M. A., and Suriadi, S. (2007). Usability and privacy in identity management architectures. In *Proceedings of the Fifth Australasian Symposium on ACSW Frontiers - Volume 68*, ACSW '07, pages 143–152, Darlinghurst, Australia, Australia. Australian Computer Society, Inc.

Kerry, C. F., Secretary, A., and Director, C. R. (2013). Fips pub 186-4 federal information processing standards publication digital signature standard (dss).

Kim, J.-J. and Hong, S.-P. (2011). A method of risk assessment for multi-factor authentication. *JIPS*, 7:187–198.

Lenz, T. and Alber, L. (2017). Towards cross-domain eid by using agile mobile authentication. In *2017 IEEE Trustcom/BigDataSE/ICESS*, pages 570–577.

Lenz, T. and Zwattendorfer, B. (2015). A modular and flexible identity management architecture for national eid solutions. In *11th International Conference on Web Information Systems and Technologies*, pages 321 – 331.

Lindell, Y. (2017). Fast secure two-party ecdsa signing. In Katz, J. and Shacham, H., editors, *Advances in Cryptology – CRYPTO 2017*, pages 613–644, Cham. Springer International Publishing.

MacKenzie, P. and Reiter, M. K. (2004). Two-party generation of dsa signatures. *International Journal of Information Security*, 2(3):218–239.

Mohammed, M. M. and Elsadig, M. (2013). A multi-layer of multi factors authentication model for online banking services. In *2013 International Conference on Computing, Electrical and Electronic Engineering (ICCEEE)*, pages 220–224.

Sarikhani, R. (2008). Language and american social identity.

Schnorr, C. P. (1990). Efficient identification and signatures for smart cards. In Brassard, G., editor, *Advances in Cryptology — CRYPTO' 89 Proceedings*, pages 239–252, New York, NY. Springer New York.

Taneski, V., Heriko, M., and Brumen, B. (2014). Password security - No change in 35 years? In *2014 37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 1360–1365.

Turner, S. (2010). The application/pkcs10 Media Type. RFC 5967.

Zwattendorfer, B. (2014). Towards a privacy-preserving federated identity as a service-model.

APPENDIX

Digital Signatures

In a nutshell, a digital signature scheme uses a message M and an asymmetric key-pair $key(sk^{sig}, pk^{sig})$ to produce a digital signature σ by using M and the private key sk^{sig} from asymmetric key-pair. A verifier can use the signature σ, the message M and the public key pk^{sig} from asymmetric key-pair to check the integrity (σ has been issued for M) and the authenticity (σ was produced by the holder of the corresponding signing key sk^{sig}) of the signature.

In a more formal way, a digital signature scheme (DSS) is a set (K,S,V) of poly-times algorithms. The first algorithm DSS_K takes a security parameter k to generate an asymmetric key-pair $DSS_K(sk^{sig}, pk^{sig})$ where the private key is sk^{sig} and the public key is pk^{sig}. The second algorithm DSS_S is the signing algorithm. This signing algorithm uses a message $M \in \{0,1\}^*$ and a private key sk^{sig} as input data and outputs a signature $\sigma = DSS_S(sk^{sig}, m)$. The third algorithm DSS_V is the verification algorithm. This verification algorithm uses the message $M \in \{0,1\}^*$, a public key pk^{sig}, and a signature σ as input data and outputs a single bit $b = RS_V(\sigma, M, pk^{sig})$, $b \in \{true, false\}$ that indicates if the signature σ is valid for M or not. Also, in a practical implementation the message M is not directly used as input data in DSS_S and DSS_V but rather $H(M)$, where H is a cryptographic hash function.

Ellipic-curve based Schnorr Signatures

Briefly, Schnorr signatures are a digital signature scheme that based on the Schnorr algorithm for identification and signature creation. The Schnorr approach was proposed by Schnorr in 1990 as a lightweight algorithm for identification and signature creation on smart cards. (Schnorr, 1990). The signing algorithm uses an asymmetric key-pair $key(sk^{sig}, pk^{sig})$ to produce a digital signature $\sigma = (R,s)$ by using an input message M and the private key sk^{sig} from asymmetric key-pair. A verifier can use the signature $\sigma = (R,s)$, the public key pk^{sig}, and the message M

In a more formal way, a elliptic curve base Schnorr digital signature scheme (ECSDSA) is a set (K,S,V) of poly-times algorithms. The first algorithm ECSDSA_K chooses an elliptic curve E over a finite field F_q. Next, the algorithm randomly selects a elliptic curve point $G \in E(F_q)$, where G is the generator in the following steps. In the last step, the algorithm ECSDSA_K takes a security parameter k to generate an asymmetric key-pair $\text{ECSDSA}(sk^{\text{ECSDSA}}, pk^{\text{ECSDSA}})$ where the private key is $sk^{\text{ECSDSA}} \in [1,r]$, where r is the order of P and the public key is $PK^{\text{ECSDSA}} = sk^{\text{ECSDSA}} \cdot P$. The full public key pk_{set} is the set (PK^{ECSDSA}, P) if the generator P was randomly chooses. The second algorithm ECSDSA_S is the signature algorithm. The signature algorithm generates random number $k \in [1,r]$, where r is the order of P. After this, a new elliptic curve point R is calculated, where $R = k \cdot P$, and P is the generator select in ECSDSA_K. In the next step, cryptographic hash value e is calculated by using a cryptographic hash function $H()$ and the message M and the point P as input data. These means that $e = H(M||R)$, where $H : 0 : 1^* \rightarrow [1,r]$ and r is the order of the generator P, and $||$ is a concatenation of M and R. At last, the signature $\sigma_M = (R,s)$ is generated, where R is the point generated before and s is calculated from $s = k + sk^{\text{ECSDSA}} \cdot e \mod r$. The third algorithm ECSDSA_V is the verification algorithm. This verification algorithm used the public key PK^{ECSDSA}, the generator R, the message M and the signature $\sigma_M = (R,s)$ and outputs a single bit $b_M \in \{true, false\}$ that indicates if the signature σ_M is valid for M, otherwise not. At first, the algorithm ECSDSA_V calculates the hash value e using a cryptographic hash function $H()$ and the message M and the point P, which is part of the signature σ_M. The single bit $b_M \in \{true, false\}$ is the proof, if $R + e \cdot PK^{\text{ECSDSA}} = s \cdot P$, where R and s are part of the signature σ_M, and P is the generator.(Schnorr, 1990; Boneh, 2005; Lindell, 2017)

Proof of Knowledge of the Discrete Log of an Elliptic-curve Point

Briefly, a proof of knowledge of the discrete log of an elliptic-curve point means the follows. For a given elliptic curve E over a finite field F_q, a generator $G \in E/F_q$, and an elliptic curve point $P \in E/F_q$, a prover wants to prove that he knows a value x such that $B = x \cdot G$, without revealing x. In respect to the elliptic curve discrete logarithm problem (ECDLP) it is very hard for an attacker to calculate a valid proof without knowing x. There exists different interactive and non-interactive zero-knowledge schemes that provide functionality to proof the possession of x (Chatzigiannakis et al., 2011; Hao, 2017). In this work, use a non-interactive zero-knowledge proof based on the Schnorr protocol; because it is lightweight, that means that it can be well integrated into smart or mobile devices. Therefore, we will give more details on this specific zero-knowledge proof.

More formally, the non-interactive zero-knowledge proof based on the Schnorr protocol consists of the following steps. In the initialize phase, a prover and a verifier agree on an elliptic curve E over a finite field F_q and a generator $G \in E/F_q$. The prover and the verifier knows an elliptic curve point $B \in E/F_q$ and the prover claims that he knows a value x such that $B = x \cdot G$. This fact should be proven to the verifier without revealing x.

1. The prover generates a random number $r \in F_q$ and computes the corresponding elliptic curve point $A = r \cdot G$.
2. The prover sends the elliptic curve point A to the verifier
3. The verifier computes a value c by using a cryptographic hash function $H()$, where $H : 0 : 1^* \rightarrow [1,r]$ and r is the order of the generator G. The value c is generated by $c = H(G,B,A)$, where G, B and A are the hash input data.
4. The verifier sends the value c to the prover
5. The prover computes the proof m as $m = r + c * x \mod q$
6. The prover sends the proof m to the verifier
7. The verifier can check that $P = m \cdot G - c \cdot B = (r + c*x) \cdot G - c \cdot B = r \cdot G + c \cdot x \cdot G - c \cdot x \cdot G = r \cdot G = A$. If $m \cdot G - c \cdot B = A$ than the prover knows x, otherwise the proof fails.

From a security point of view, a dishonest prover has a tiny chance for cheating as he would have to fix the value of $P = m \cdot G - c \cdot B$ before receiving the hash value c from the verifier. However, under the assumption that the cryptographic hash function $H()$ is secure, a prover that does not know x cannot cheat in respect to the discrete logarithm of B.