Attributes and tokens in U-Prove: Interval proofs and use cases

Pieter Rogaar

August 14, 2011

ABSTRACT

People wish to control their private information when interacting with computer systems. At the same time, online service providers ask for authenticated and more detailed information about their customers. Microsoft has published U-Prove, a cryptographic system for the selective disclosure of authenticated attributes. In this system, a user can retrieve a token from a party which third parties trust to authenticate the user's attributes. This token, which is essentially a piece of digital data, encodes the attributes of the user, for example, his name, date of birth or access authorizations. Using this token, the user can present some of this information to a service provider, who can verify that the identity provider has indeed issued the token. In this thesis, we have added to this functionality by introducing a new type of proof, with which the user can show that the value of one of his attributes lies in a given interval. An example would be the claim "I am between 18 and 65 years of age."

In this thesis, we also have researched the factors which are relevant to the potential success of U-Prove. We have studied three separate use cases for the U-Prove system: the Dutch public transport card OV-chipkaart, the Dutch electronic identity card eNIK and the Dutch Ministry of Defense identity card Defensiepas. In each of these cases, we have identified the relevant stakeholders in the debate surrounding the introduction of such a privacy enhancing technology. After presenting their views, we have described the expected difficulties which might arise when one tries to implement U-Prove in each of these cases. We furthermore have formulated three views on the way in which U-Prove might be incorporated in the development trajectory of a new or existing system. We can add U-Prove as an afterthought, in which case we only profit in a limited measure from its privacy guarantees, we can redesign our system architecture with U-Prove in mind, which would result in a much more privacy-friendly system, but with large impact on development costs, or we can reform our whole idea about identities around the notion of pseudo-identities and selective disclosure which U-Prove propagates. This requires not only an organizational, but also a philosophical change in the way an organization handles the identities of its users.

It is our choices that show what we truly are, far more than our abilities.

Albus Dumbledore
in *Harry Potter and the Chamber of Secrets*
J.K. Rowling

CONTENTS

1. FOREWORD

Πάντα στον νου σου νάχεις την Ιθάκη.
Το φθάσιμον εκεί είν' ο προορισμός σου.
Αλλά μη βιάζεις το ταξείδι διόλου.
Καλλίτερα χρόνια πολλά να διαρκέσει·
και γέρος πια ν' αράξεις στο νησί,
πλούσιος με όσα κέρδισες στον δρόμο,
μη προσδοκώντας πλούτη να σε δώσει η Ιθάκη.

From *Ithaka*, K.P. Kavafis

This thesis marks the end to a diverse study trajectory. I believe that it embodies all three of my primary academic interests: cryptography, computer security and the real world. That implies that you, the reader, will probably find it either too technical or not rigorous enough. This is no matter: It was meant to be that way.

I heartily hope that this thesis will contribute to the success of cryptographic privacy enhancing technologies, as these have the potential for empowering users in their online identity management. Lately, I have perceived an increased interest in the subject. On the one hand, people have finally realized that no one but they can effectively and conscientiously manage their online identities. On the other, people are anxious to find technologies and techniques which enable them to keep their online privacy intact without limiting the spectrum of activities they can employ online.

I obviously did not do all this work alone. I would like to thank my supervisors, Pieter Ceelen, Pim Vullers and Bart Jacobs, for their support, their experience and their good ideas: I'm proud that I may present many of these with my name on the front cover.

Nobody knows better what their respective organizations could gain from a technology like U-Prove than the people I have interviewed: Sergej Katus, Fekke Bakker and Bas van Os, thank you.

Without the suggestions of Berry Schoenmakers, the mathematical portion of this thesis would be a mere rehashing of work of a decade ago.

Many others have contributed their valuable professional opinions and knowledge on the subject of online privacy and identity management: I heartily thank

Patrick Paling, Rachel Marbus, Joris ter Hart, Ronald Koorn, Daniel Wunderink, Marcel Wendt and Christian van 't Hof for the thousand new ways in which I learned to look at my own research subject.

If there is one thing I have learned during these last few months, it is that I am unable to be alone: I need people who value and support me. Therefore, I thank all my friends and family: without you, not only would most of this thesis have remained unwritten, but I also would not be the person I am today, the one who was both capable and curious enough to write it.

I have written this thesis during my internship at KPMG IT Advisory, IT Security and Control. I am thankful for the opportunities they have already given me, as these were instrumental to writing this thesis. I am looking forward to the future, as I will grow into my role of advisor at this firm.

2. INTRODUCTION

> Freedom is the freedom to say that two plus two make four. If that is granted, all else follows.
>
> *Nineteen Eighty-Four*, George Orwell

Individuals wish to control their personal information in the online domain. Organisations who are responsible for handling the information of individuals seem to be minimally concerned with this wish, as can be seen from the large number of severe data leaks during the past years. One guiding principle, data minimalization, is hardly ever practiced and almost never enforced, which leads to very limited user empowerment with respect to privacy. On the other hand, the service providers which rely on the personal data of their users are asking for more accurate and detailed information, preferably authenticated by a trusted party such as the government.

In order to help users to selectively disclose their authenticated personal information, Microsoft has published U-Prove, an open cryptographic specification which can be used in the online domain. U-Prove allows users (e.g citizens or customers) to disclose information about themselves which has been previously authenticated by a trusted party such as the government, but to do so in a controlled manner. For example, if the government authenticates all the information in your passport, you will be able to show only your first name and date of birth to a third party. Showing this information occurs without any involvement from the government.

The U-Prove specification allows for some quite strong privacy guarantees. In the light of the current situation, implementing U-Prove broadly would be a significant step forward for end-user privacy in many domains. The question therefore is: why is this not happening yet? U-Prove was published in March of 2010, and as of yet, we are unaware of any implementations of U-Prove in production systems.

While the mathematical ideas behind U-Prove were already published a decade ago, the technology itself is still in its infancy. We will study the different factors behind the potential success of U-Prove. Our main research question therefore is:

> What are the driving and hampering factors behind the successful adoption of U-Prove in existing or newly designed systems?

We answer this question by researching several subquestions, namely:

- How, in broad lines, does U-Prove work and what privacy and security guarantees does it offer?
- How can U-Prove be extended to support more advanced presentation proofs, such as interval proofs?
- What are some of the potential use cases of U-Prove, and what specific aspects of those are relevant to the adoption of U-Prove?
- What commercial, societal, legal and technical driving and hampering factors for the adoption of U-Prove exist?

We have approached this research question in several ways. First, we have studied U-Prove and the mathematics behind it. We have formulated an extension to U-Prove which allows for the disclosure of interval containment of attributes, and we have implemented this extension in the context of the experimental U-Prove SDK which was published by Microsoft. Next, we have performed three case studies of systems where U-Prove could potentially be implemented: we have studied the OV-chipkaart, eNIK and Defensiepas systems by conducting interviews with involved parties. We have summarized some of the potential approaches to implementing U-Prove in these cases and identified the conflicts which might arise. We distilled the driving and hampering factors for the adoption of U-Prove from these case studies and have drawn some conclusions about general approaches towards implementing U-Prove.

Lay-out of this thesis After this introduction, Chapter 3 presents a basic introduction to the functionality of U-Prove and the terms which are used when discussing implementations of U-Prove. All further chapters require only this amount of knowledge, though the reader will appreciate some subtleties more readily after having also perused Chapter 4, which presents a technical description of U-Prove and its functionality. Chapter 4 also contains pseudo-code for the algorithms which are used for interval proofs. In Chapters 5, 6 and 7, we respectively discuss the case studies of OV-chipkaart, eNIK and Defensiepas. Chapter 8 summarizes and generalizes our findings of the previous chapters and discusses several high-level approaches to applying U-Prove to another project. The appendix contains derivations of the interval proof presentation and verification algorithms, as well as a brief description of the modifications we have made to the U-Prove SDK source code.

3. NON-TECHNICAL OVERVIEW OF U-PROVE

> All this happened, more or less.
>
> *Slaughterhouse-Five*, Kurt Vonnegut

In this chapter, we will briefly introduce U-Prove and the most important concepts of this technology. We will explain the roles which exist: Issuer, Prover and Verifier. We will also describe the life cycle of a U-Prove token, and touch upon the different proofs which can be generated with such a token.

3.1 Introduction to U-Prove

U-Prove is a cryptographic technology which is owned by Microsoft. Originally, it was invented by Stefan Brands in his PhD thesis [5]. Microsoft has released U-Prove under its Open Specification Promise, which implies that anyone is free to use, implement or modify the specification without risking any claims by Microsoft due to patent infringement. However, no assurance is given with regard to patent claims by third parties. Furthermore, an experimental SDK has been released (C# and Java) to enable developers to experiment with the technology themselves.

In one sentence, U-Prove is a system which allows for *selective* and *off-line* disclosure of *authenticated* attributes. It is an alternative to an identity based authentication system. The attributes are issued by a trusted party and signed upon issuance, so that a relying party may *authenticate* them. The disclosure of these attributes can happen long after the interaction with the issuing party has ended, and can therefore be performed *off-line*. Also, not all attributes need to be shared upon every presentation: the party which shows the attributes can *select* which attributes he wishes to disclose. Essentially, U-Prove is a manifestation of the definition of privacy as it is given in [29]:

> [Privacy is] the claim of individuals, groups, or institutions to determine for themselves when, how, and to what extent information about them is communicated to others.

The system centers around three roles: Provers, Issuers and Verifiers.

- A Prover is a party which has certain attributes (name, phone number, date of birth, access authorizations, etc.) which he wants to disclose to Verifiers. He is not trusted himself to attest the validity of these attributes - he will need to ask a party which Verifiers trust, an Issuer, to sign his

set of attributes. He does not trust the Issuer fully: he does not want to disclose with which Verifiers he interacts. He also does not trust Verifiers fully: he only wants to disclose the attributes which are relevant to his interaction with that particular Verifier.

- An Issuer is a party which Verifiers trust to attest the validity of attributes of Provers. He will use his digital signature to show that he indeed agrees with the attributes which the Prover presents. He does not trust Verifiers with any information which the Prover does not wish to disclose. He also does not trust Provers to be honest about the attributes they disclose. He can not actively influence which Prover shows his attributes to which Verifier.

- A Verifier is a party which relies on an Issuer to attest the validity of attributes of a Prover. He does not trust the Prover to show valid attributes, but he does trust the Issuer to only sign valid attributes. He is curious about the attributes which an Issuer signed, but which the Prover does not want to disclose. An Issuer can also play the role of Verifier. For example, a government institution which authorizes certain identity elements on an identity card of a citizen might use this same identity card to verify some of these identity elements.

An example of a context in which U-Prove may be used is showing a passport. Traditionally, a passport contains many different attributes about your identity, such as your name, date of birth, social security number and photograph. When you show your passport to somebody, you immediately disclose all these attributes to this party. If your passport was based on U-Prove, it would contain your personal information in attributes which were signed by the government, the Issuer, and which you, the Prover, can selectively disclose to a relying party, which is the Verifier. For example, you can choose to show only your first name and your photograph, but no other information. Additionally, a passport is produced in such a way that a party which checks your passport can verify its validity. The attributes which you use with U-Prove are also signed, so you can show that the government has issued them, just as with the regular passport. Finally, another parallel between a regular passport and the one with U-Prove is that in both cases, the Issuer (the government) does not find out that you have presented the information in your passport to a Verifier.

3.2 U-Prove tokens and proofs

To achieve the functionality which we described in the example in the previous paragraph, U-Prove contains the notion of a *token*. A token is a piece of data which contains a set of attributes of a Prover and which is signed by the Issuer. We will briefly discuss two elements in the life cycle of a token: issuance and presentation, and elaborate on the two possible presentations in a bit more detail.

- *Issuance:* A token is constructed in a protocol between the Issuer and the Prover, which explicitly does not involve any Verifier. Using a cryptographic technique called blind signing, the Prover makes sure that the

Issuer does not know the (unique) Token Identifier which is included in every token. The Issuer obviously needs to know the attributes included in the token, as he attests their validity.

- *Presentation:* A Prover can decide to present some of the attributes which are stored in a token to a Verifier. He discloses these attributes to the Verifier, after which he proves with a zero-knowledge protocol[1] that the token he possesses (and which is signed by the Issuer) indeed contains the attributes which he just disclosed. In this protocol, the values of the attributes which he did not disclose remain hidden to the Verifier. In this interaction, the Verifier notes the Token Information field of the token, which enables him to recognize a token which has been presented to him before. Note that the Issuer is explicitly not involved in this interaction (except when he plays the role of Verifier himself). The Verifier obtains a transcript which he can use to prove that he indeed has interacted with a Prover which did possess the required attributes.

- *Interval presentation:* The current version of U-Prove allows for selective disclosure of attributes. However, on a per-attribute basis, this is an all-or-nothing affair: if one of the attributes is your date of birth, you can not decide to disclose just the year in which you were born. Therefore, we present a technique in this thesis which allows for *partial disclosure* of an attribute, namely by proving that the value of one of the attributes lies in a given interval. From a high level, this technique is comparable to the regular presentation proof: the Prover generates a proof with a zero-knowledge protocol that the attribute in the token he possesses indeed lies in the given interval. The Verifier can then check this proof to ascertain himself that the value indeed lies in the given interval. A typical example of such an interaction is provided in Figure 3.2.

3.3 Security properties of U-Prove

- U-Prove provides a (temporal) separation between the interaction between Prover and Issuer and the interaction between Prover and Verifier: after the Issuer issues the token, there is no way for the Issuer to find out with which Verifier the Prover chooses to interact. This provides strong privacy guarantees for the Prover. Not even if the Issuer later plays the role of a Verifier can he recognize a Prover with whom he has previously interacted.

- Even if a token contains sensitive information, a Prover can selectively disclose just those attributes which are required for the interaction with the Verifier. The Verifier has no way to determine the values of the other attributes.

- Through the Token Information field, it is possible for a Verifier to recognize a Prover with whom he has previously interacted. This is required, for example, in shops which wish to monitor customer relations. Note that

[1] A zero-knowledge protocol is a cryptographic protocol in which one party (the Prover) shows to another party (the Verifier) that he has certain knowledge without actually disclosing this knowledge to the other party. For more information, see Section 4.4.

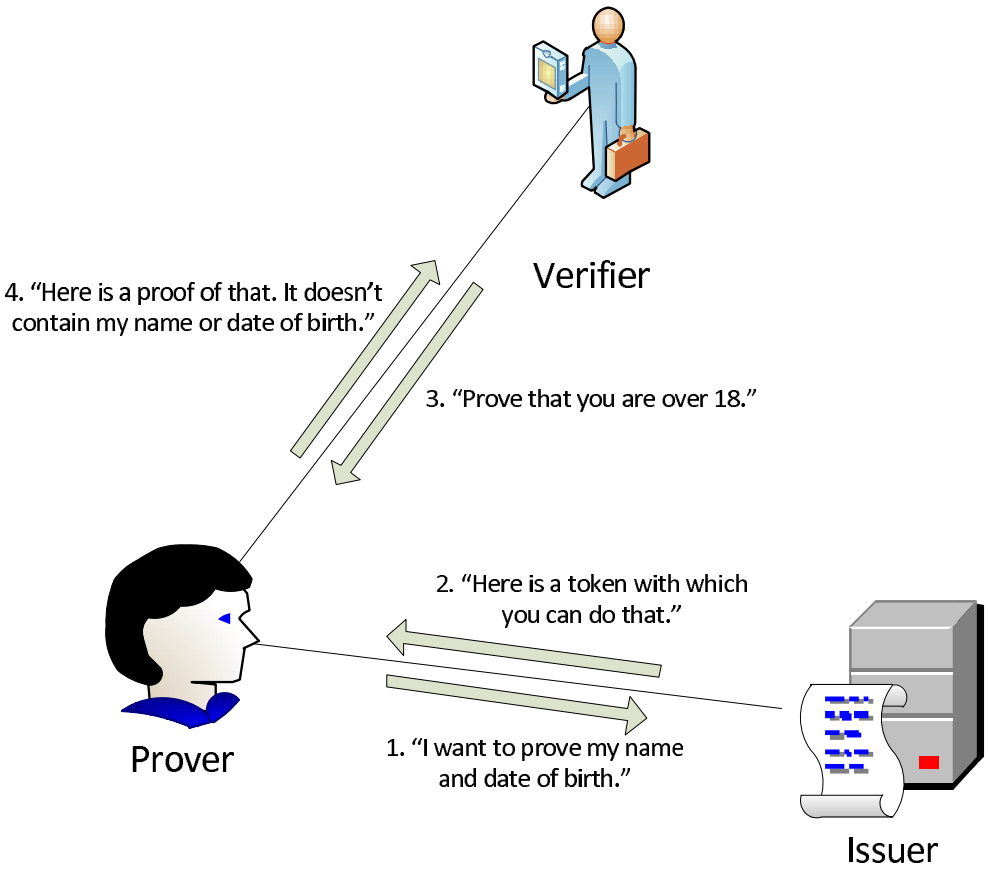

Fig. 3.1: The interactions between Prover, Issuer and Verifier when performing a partial disclosure of an attribute. Note that steps 1 and 2 can occur long before 3 and 4.

a Prover can always decide to request a new token from the Issuer which contains the same attributes. Using this new token instead of the old one, he can establish a new 'pseudo-identity' with a Verifier.

- If a Prover uses the same token to interact with two different Verifiers, these Verifiers can use the transcript of their conversation with the Prover to determine that they have communicated with the same Prover. If the Prover uses two different tokens for these interactions, the Verifiers have no way of discovering that these interactions were, in fact, with the same Prover.
- The storage and computations concerning the issuance and presentation of a U-Prove token can be performed by a trusted computing device such as a smartcard. However, the limited computational capacity of a smartcard implies that the computations can not be performed quickly enough in some scenarios. For example, disclosing one of five available attributes took 0.8 seconds on a common smartcard, but this time increases quickly when the total number of attributes increases [9].

3.4 Future properties

- Revocability of attributes is an important requirement for real-life applications. Currently, U-Prove does not support revocability. It is expected that future releases will contain cryptographic measures to achieve this. Both the specification of Microsoft [10] and the original description of U-Prove by Brands [5] contain suggestions for revocation mechanisms, but these are exceedingly slow when applied straightforwardly. For example, Microsoft suggests including a unique 'secret' attribute in the token, which is never directly disclosed. Rather, a zero-knowledge proof is constructed which shows that this value is *not* equal to a certain revoked value. This mechanism requires a great number of zero-knowledge proofs: straightforwardly proving that the value of the secret attribute is equal to one of the non-revoked values requires a large OR-construction. Such a construction requires $\mathcal{O}(n)$ presentation proofs, where n is the number of non-revoked values.
- The currently released version of U-Prove only allows for disclosure of the attributes themselves. In this thesis, we present an extension for U-Prove which allows the presentation of derived attributes (e.g. 'date of birth' (fully disclosed) vs. 'over 18?' (derived attribute)).
- More expected features of U-Prove can be found in the U-Prove Technology Overview [11].

3.5 Summary

In this chapter, we have described the aspects of U-Prove which should be understood before reading the rest of this thesis. To summarize: U-Prove is a privacy-friendly alternative for systems that deal with identities and/or attributes. A U-Prove token is issued by an identity provider, which we call an Issuer. After this issuance, the owner of the token, the Prover, can use it to

selectively disclose some of the attributes which are encoded in it to a third party, a Verifier. This disclosure happens off-line, which means that the Issuer does not learn that the attributes are disclosed. Also, the Verifier can check the signature of the Issuer to verify that the encoded attributes are authentic.

4. TECHNICAL EXPOSITION ON U-PROVE

> Let every eye negotiate for itself,
> And trust no agent.
>
> *Much Ado About Nothing*, William Shakespeare

In the previous chapter, we introduced the functionality of the U-Prove technology. In this chapter, we will describe the mathematics behind this technology. Since the issuance and presentation protocols are based on the concept of a zero-knowledge proof, we will explain some of the core concepts of this notion. Furthermore, we will touch upon the concept of a blind signature, as such a signature is used in the issuance protocol to assure the privacy of the Prover with respect to the Issuer.

4.1 U-Prove functionality

As mentioned in the previous chapter, U-Prove enables a Prover to present attributes from a token which the Issuer has assigned to him. The party to whom these attributes are presented, the Verifier, needs only to trust the Issuer to assign the correct attributes to the Prover. On the other hand, the presentation happens off-line: the Issuer is not informed of the fact that the Prover uses his credentials or to which Verifier he presents them. Some examples of attributes for which use cases can be easily envisioned are the Prover's name, e-mail address, bank account number, date of birth, health insurance company and number, military rank and building access authorizations. For each of those, Verifiers exist which require this data in an authenticated form. On the other hand, most Verifiers should not be able to access all attributes. Futhermore, a Prover often does not want the Issuer to know to which Verifier he presented his authenticated attributes, or even that he did so at all. Even if the Issuer functions as a Verifier after issuing a token, he should not be able to link that token to its issuance when it is presented to him. The U-Prove technology provides a method to achieve all of these goals.

There are three types of fields in a token which can contain information which is specified at issuance time. Each of those types has a specific role in the guarantees that U-Prove provides.

- The token information field contains a value TI which is encoded by the Issuer that is always disclosed when the Prover presents the token. Typical uses of this field include information about expiration dates or other metadata.

- The Prover information field contains a value PI which is encoded by the Prover. It is kept secret from the Issuer, but it is always presented when the token is used.
- The attribute fields contain values $(A_1, \ldots, A_n)$ encoded by the Issuer. These fields can be selectively hidden or disclosed by the Prover when he uses his token.

The life cycle of a U-Prove token has three stages. In the first stage, the issuance of the token is achieved through the issuance protocol between the Issuer and the Prover. After completing this protocol, the Prover holds the token which he can use in the second stage. In this stage, he uses the presentation protocol to disclose some or all of the attributes which are included in the token to a Verifier. He can do this repeatedly and with many different Verifiers. In the third stage of the life cycle of a token, the Issuer wishes to prevent the Prover from subsequently using the token. Note that Microsoft did not explicitly include a revocation mechanism in the U-Prove specification, but they did suggest several ways in which to achieve revocation [12], such as proving that a special attribute value is *not* on a given revocation list. Furthermore, an Issuer may decide to encode an expiration date for the token in some pre-agreed manner into the token information field. After this date, Verifiers should no longer accept the token. However, this is not part of the U-Prove specification.

4.2 Mathematical context

The original formulation of U-Prove [5] describes two possible representations for the attributes and the token, which lead to two different underlying security assumptions. The first is the discrete logarithm representation - in that case, the security guarantees stem from the discrete logarithm (DL) assumption [5]. The second is the RSA representation, which depends on the RSA assumption. As the subsequent work of Microsoft [10] focuses on the DL representation, we will consider only this representation in this thesis. Note that an actual implementation was made of the Prover algorithms for U-Prove on a smartcard, which was based on the RSA represenation [9].

In order to ensure interoperability between Issuers, Provers and Verifiers, the public parameters of the mathematical operations are encoded in an Issuer parameters instance, which is of the following form:

$$\mathrm{UID}_P, (p, q, g), \mathrm{UID}_{\mathcal{H}}, (g_0, g_1, \ldots, g_n, g_t), (e_1, \ldots, e_n), (z_0, z_1, \ldots, z_n, z_t), S$$

where

- UID_P is an octet string which is a unique identifier of the Issuer parameters within the application context.
- $p, q \in \mathbb{Z}$ and $g \in \mathbb{Z}_p$ with p and q prime, $q \mid p-1$, such that Z_p is a field for which the DL assumption holds and G_q, the subgroup of Z_p which is generated by g, is of order q.
- $\mathrm{UID}_{\mathcal{H}}$ is an identifier of the secure cryptographic hash function to be used. The values it outputs can be easily mapped to $\mathbb{Z}_q$.

- $(g_0, g_1, \ldots, g_n, g_t) \in (G_q)^{n+2}$ is the Issuer's public key. This public key corresponds to a secret key $y_0 \in G_q$, which should be kept secret by the Issuer.
- $(e_1, \ldots, e_n)$ is a list of byte values which indicate whether the corresponding attribute values $(A_1, \ldots, A_n)$ should be hashed before calculating the public key h of a token. If e_i equals `0x01`, A_i should be hashed. If e_i equals `0x00`, A_i should not be hashed.
- $(z_0, z_1, \ldots, z_n, z_t)$ are values which are used by the Prover in the issuance protocol.
- S is an octet string which contains a specification of the Issuer parameters and the tokens that are issued with them.

The Issuer parameters can, for example, be stored in a centralized location, with the different participants of the protocol only exchanging the UID_P string and retrieving the corresponding parameters from this location.

4.3 Structure of a token

A U-Prove token is a piece of data which enables a Prover to prove to a Verifier that the Issuer has agreed to the attributes which are stored in the token. The public part of a U-Prove token is of the following form:

$$\text{UID}_P, h, TI, PI, \sigma'_Z, \sigma'_c, \sigma'_r$$

where

- UID_P is the identifier of the Issuer parameters which were used when issuing this token.
- $h \in G_q$ is the public key of the U-Prove token. See below for details on this public key.
- $TI \in \{0,1\}^*$ is the value of the token information field. This field is used to store information which the Issuer encoded and which should always be disclosed to Verifiers when using the token.
- $PI \in \{0,1\}^*$ is the value of the Prover information field. This field is used to store information which the Prover encoded into the token and which should remain secret from the Issuer. It is always disclosed to the Verifier when a token is used. Example uses are including an encryption key or a nonce to guarantee freshness of the token.
- $\sigma'_Z \in G_q$ and $(\sigma'_c, \sigma'_r) \in (\mathbb{Z}_q)^2$ form the Issuer's signature on all the other token contents. By verifying this signature, the Verifier assures himself of the agreement of the Issuer with the token contents.

All the previously mentioned values in the token are disclosed when presenting the token. However, there are also two types of values which are kept secret:

- The private key of the token $\alpha^{-1} \in \mathbb{Z}_q^*$, which is the multiplicative inverse to a randomization value $\alpha \in \mathbb{Z}_q^*$ which is generated by the Prover during the issuance protocol.[1]
- The values $A_1, \ldots, A_n \in \{0,1\}^*$ of the attributes which are encoded into the token. These need not be specific to the token. In a typical scenario, multiple tokens will exist which are used to present the same attribute values. This enables a Prover to manage multiple 'pseudo-identities', because these tokens can not be linked except by the attribute values. Since the values of the disclosed attributes are known in any case, this provides for the highest attainable measure of anonymity. Note that the attributes that are shown to a Verifier are, of course, no longer kept secret from that Verifier.

Token public key

Every U-Prove token contains a public key $h \in G_q$. The general form of such a public key is

$$h = (g_0 g_1^{x_1} \ldots g_n^{x_n} g_t^{x_t})^{\alpha} \bmod p$$

where

- $(g_0, g_1, \ldots, g_n, g_t) \in (G_q)^{n+2}$ is the Issuer's public key, which is taken from the Issuer parameters.
- $\alpha \in \mathbb{Z}_q^*$ is the multiplicative inverse of the token private key.
- $x_t \in Z_q$ is a value which is computed by hashing the version number of the issuance protocol, a hash of the Issuer parameters under which the token was issued and the token information field. The presentation protocol treats this field differently from the other attribute fields: for one, this attribute is always hashed. Also, the Prover can not choose whether to show this attribute: it is always disclosed when a proof is constructed.
- $x_i \in \mathbb{Z}_q (1 \leq i \leq n)$ is computed from the corresponding attribute value A_i either by encoding it directly (if the value e_i in the Issuer parameters equals `0x00`) or by hashing it first (if e_i equals `0x01`). In the former case, the bit string A_i is interpreted as an integer, which should be less than q in order to be interpreted correctly as an element of $\mathbb{Z}_q$.

The public key is generated during the issuance protocol. The public key is never seen by the Issuer, and the Issuer can not correlate the disclosed public key in a presentation protocol instance to the instance of the issuance protocol in which the corresponding U-Prove token was issued.

[1] It is possible to deterministically generate the token private keys from an existing private key, such as one which is stored in a smartcard. Preventing the sharing of tokens should, however, not rely on such a feature, as other mechanisms to achieve this are in place. For example, the Issuer could encode an attribute in the token which the Prover will never want to share. Since the Prover needs to know all the attributes to correctly perform the presentation protocol, sharing a token with another Prover would imply that the Prover needs to disclose this attribute to this other Prover.

Fig. 4.1: Wally, the protagonist of our first zero-knowledge protocol.

4.4 Intermezzo: Zero-knowledge proofs and blind signatures

In the previous sections, we have introduced the mathematical context of the U-Prove technology. Two important questions have, however, been left unanswered: First, how do Prover and Issuer generate a token and second, how does a Prover present the attributes for which a token was issued to a Verifier? To answer these questions properly, we first provide a brief introduction into the field of zero-knowledge protocols. Using such protocols, parties can prove possession of certain knowledge or computational abilities, while sharing only an absolute minimum of information with the other party. We first present two examples of zero-knowledge protocols: the non-technical 'Where is Wally?'-protocol and the Schnorr identification protocol. Next, we discuss the concept of blind signatures, as this is used while issuing U-Prove tokens. It follows naturally from our initial exposition on zero-knowledge protocols.

Where is Wally?

Many are familiar with the children's game 'Where is Wally?'. On a poster-size drawing, Wally, a comic character who is always dressed in a red and white striped sweater and beanie, is hidden among many other people. The busy nature of the drawing and the many other characters who are dressed in similarly coloured clothes make it difficult to pinpoint Wally's location.

Imagine two players, Alice and Bob, are playing 'Where is Wally?' on a given drawing. Alice has found Wally on the drawing and informs Bob of this fact. Bob does not believe her - after all, he has not been able to find Wally yet, so for all he knows, Wally might not even be depicted on the drawing. Alice wants to convince Bob that she has actually found Wally. However, if she points out Wally to Bob, Bob will no longer be able to attempt to find Wally himself, which would ruin the game for him. Every technique for directly proving to Bob that she knows Wally's location either gives Bob some additional information about Wally's location or does not convincingly prove to Bob that she indeed does know Wally's location. After a while, she finds the following technique for proving her knowledge to Bob [13]:

'Where is Wally?'-protocol Alice takes a large piece of cardboard (at least twice the size of the drawing on which she has found Wally) and uses it to cover the drawing in some arbitrary orientation, but with Wally not too close to the edge of the cardboard. Now, she shows this setup to Bob. Note that Bob does not even yet know whether the drawing on which he was looking for Wally earlier is under the cardboard. Alice asks Bob: "Uncover or show?". If Bob chooses uncover, Alice removes the cardboard, and Bob can ascertain himself that the drawing which they perused earlier was indeed under the cardboard. If Bob chooses show, Alice pokes a small hole in the cardboard at the location where Wally is shown in the underlying drawing. Bob can now see Wally through the hole, though he can not see the surroundings of Wally, so he knows nothing more about the location where he is hidden.

It is easy to see that Alice can cheat in the previous protocol if she knows beforehand what option Bob will choose. If he chooses uncover, she just places the drawing which they studied below the cardboard without knowing where Wally is. If he chooses show, she takes a different image of Wally and puts that below the cardboard at the location where she will poke a hole. Since Alice does not know Bob's choice beforehand, she can cheat with a probability of $\frac{1}{2}$. It is crucial to the effectiveness of the protocol that Alice first *commits* to an instance (the setup of the drawing with the cardboard on top), after which Bob formulates a *challenge* for Alice ("Uncover or show"). Next, Alice needs to perform the proper *response* action (either uncover the drawing or poke a hole at Wally's location). If Alice does not know the location of Wally, and Alice and Bob iterate this protocol k times, the probability that Alice will not be caught is 2^{-k}. Therefore, Bob can become arbitrarily certain that Alice has indeed found Wally. At the same time, note that the conversation contains no information which is new to Bob: he already knows what Wally looks like, and he also knows what the drawing which is hidden looks like. It is exactly this property which is captured by the phrase 'zero-knowledge'.

Schnorr's identification protocol

The exposition in the previous paragraph has illustrated some key concepts from the field of zero-knowledge protocols. To familiarize ourselves some more with the way in which zero-knowledge protocols work, we present a mathematical example, namely Schnorr's identification protocol [17]. Using this protocol, a proving party can prove to a verifying party that he knows the discrete logarithm of an element in a finite field.

Let $G = \langle g \rangle$ be a group of order p, with p a large prime. Let $x \in \mathbb{Z}_p$ be Alice's private key, and $h = g^x \in G$ be the corresponding public key. Suppose that Alice wishes to demonstrate to Bob that she has the private key which corresponds to g^x. In order to prove this, Alice and Bob participate in k rounds of the protocol in Figure 4.4, where $k \in \mathbb{Z}_{>0}$ is a parameter which expresses the degree of certainty Bob has about Alice's knowledge of x. More precisely, the probability that Alice is cheating undetectedly is less than or equal to 2^{-k}.

In this protocol, we see the same three-step process which was used in the 'Where is Wally?'-protocol. First a commitment a is generated and sent by Al-

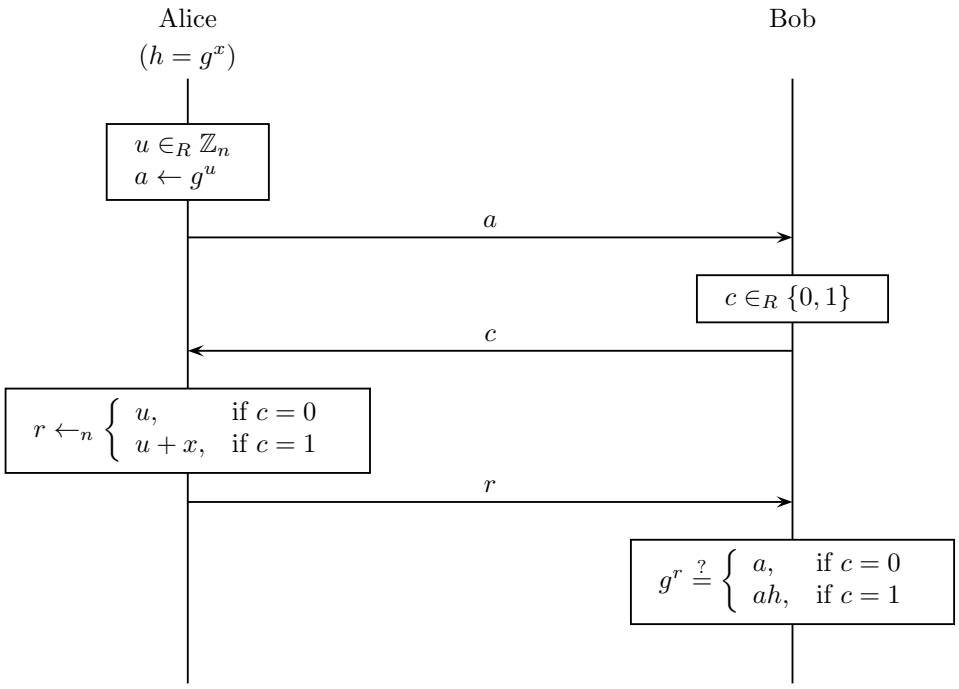

Fig. 4.2: Schnorr's identification protocol

ice. Bob now generates a challenge c, which he returns to Alice. Alice now needs to compute a response r to Bob's challenge in the context of her commitment.

We first discuss why this protocol indeed convinces Bob that Alice knows $x = \log_g h$. This property is named the *soundness* property. Note that if Alice is cheating (i.e. she does not know x), the best she can do is to prepare for either the case $c = 0$ or $c = 1$. If $c = 0$, she should prepare by calculating and sending the commitment in the normal way, as she is not asked to include any information about the value of x. She can then send the proper response without knowing x. If $c = 1$ however, she should prepare by calculating the commitment normally, but not sending $a = g^u$, but rather $a/h = g^{u-x}$. Now, in the third step of the protocol, she sends $(u - x) + x = u$ to Bob, who accepts this response.

The point is that Alice can only properly respond to both the $c = 0$ and the $c = 1$ case if she knows $x = \log_g h$. To see why this is the case, consider the following. Suppose that Alice can respond properly to both challenges. Now, after she has sent a to Bob, which she has calculated in some arbitrary but efficient way, Bob has sent her a challenge $c \in \{0, 1\}$. Since she knows both answers which Bob expects, she apparently has access to some r_1 for which $g^{r_1} = a$ holds, as well as some r_2 for which $g^{r_2} = ah$ holds. Now, note that $g^{r_2 - r_1} = h$, so $r_2 - r_1 = x \bmod n$. This proves that if Alice can answer correctly to both challenges, she can efficiently compute x. Therefore, no cheating Alice can answer correctly to both challenges[2].

To see why Schnorr's identification protocol is *zero-knowledge*, we briefly note that Bob could have generated the conversations (a, c, r) which he has with Alice on his own as well, without any interaction with Alice. In order to do this, he first randomly picks a challenge $c \in \{0, 1\}$. He then generates $r \in_R \mathbb{Z}_n$. Last, he calculates $a = g^r h^{-c}$, after which he outputs (a, c, r). Note that the set of simulated conversations has the same distribution as the set of conversations which would arise from normal protocol executions. Therefore, showing such transcripts does not in any way prove that interaction with Alice took place. One additional remark is in order: this only covers the case of an honest Bob, who chooses c uniformly from the set $\{0, 1\}$. A curious Bob might want to know the actual value of x, and try to cheat in order to find x. He can try to cheat by using a different probability distribution for c, but this will also not help him: he is still not able to retrieve x. Proving this in the case of a cheating Bob is handled in [19].

Note the similarities between the two protocols we have discussed so far. In particular, mark the correspondence between the mathematical properties we have discussed in this subsection and the intuitive constructions of the previous subsection: the 'uncover' challenge corresponds to $c = 0$ in Schnorr's protocol, and the 'show' challenge corresponds to $c = 1$ in Schnorr's protocol. In the next

[2] Note that this exposition only shows that Alice must hold the private key if she can answer properly with probability 1. However, it is also true that if she does not have the key, she can only answer properly with probability $\frac{1}{2} + \varepsilon$, with ε negligible in the security parameter. [17]

subsection, we will discuss an application of zero-knowledge protocols: blind signatures.

Blind signatures

In the non-digital world, signatures are regularly used to show that a party agreed to, approved of or read something: E.g. the contents of a contract, a petition or the report card of your child. Even though these signatures are relatively easy to forge, they remain in active use. The digital world has a cryptographic equivalent, the digital signature. Through the use of a digital signature scheme, one may attach an identifiable, verifiable and unforgable data element to a message, which we call the digital signature. Many digital signature schemes are based on public key cryptography. In recent years, these digital signatures have even been included in legislature, giving them a legal status which is equivalent to non-digital signatures. [25]

In typical application scenarios of digital signatures, one party functions as a signing authority. This party can sign messages to signify that he approves of the contents or to show that he believes some other property of the message to hold, such as authenticity or freshness. Any party can now verify the signature which the signing authority has placed on the message. However, in some cases, it is necessary that the message does not become known to the signing authority. A typical application where this is the case is with electronic cash: in order for parties to trust the currency data units, these need to be signed by a signing authority, such as the issuing bank. On the other hand, the signed currency data units should not be traceable to some specific issuance, so as to preserve the anonimity of the party which spends the currency. In other words, it should be impossible for the bank to correlate a currency data unit it receives to one it has previously issued. To this effect, the receiving party usually obfuscates the message which it requests to be signed by the signing party. After the signature has been applied, the receiving party removes the obfuscation[3] to be able to use the combination of message and signature freely.

An example of a blind signature scheme is the one introduced by Chaum [6]. Chaum's blind signature scheme is based on digital signatures with the RSA cryptosystem, which we will briefly describe here. In this system, a public key consisting of a public modulus $m = pq$ and a public exponent e with $\gcd(e, \phi(m)) = 1$. The private key consists of the private exponent $d = e^{-1} \bmod \phi(m)$ and the prime factors p, q of the public exponent m. A signature on a message M is generated by applying a secure cryptographic hash function $\mathcal{H}$ to M and subsequently applying the secret exponent to the result: $S = \mathcal{H}(M)^d$. Verifying the signature happens by applying the public key and verifying that $\mathcal{H}(M) = S^e \bmod m$.

In the system by Chaum, the hash is calculated by the receiver of the signature. He then multiplies by an obfuscation factor r^e, and sends the result

[3] This is usually made possible through some mathematical property of the underlying (public key) cryptosystem, such as a homomorphic property.

$\mathcal{H}(M)r^e$ to the signer. The result now is equal to

$$(\mathcal{H}(M)r^e)^d,$$

which is equal to

$$\mathcal{H}(M)^d r^{de} = \mathcal{H}(M)^d r = Sr.$$

The receiver of the signature knows the value r and can therefore retrieve the actual signature S. Further details on the scheme by Chaum can be found in [6].

4.5 Issuing a token

After the previous intermezzo, we now return to the U-Prove context to describe the procedure to issue a token. The first step in the life cycle of a token is its issuance: the Issuer and the Prover interact using the issuance protocol to generate an Issuer signature on the contents of the token. Since some of the contents of the token should remain hidden from the Issuer, the signature scheme will be partly based on the blind signature principles which we introduced in the previous section.

First, the Issuer and the Prover need to agree on most of the contents of the token, namely the Issuer parameters, the attributes $(A_1, \ldots, A_n)$ to be encoded in the token and the value of the token information (TI) field. How this happens is not within the scope of the U-Prove framework, but in most cases this will require some kind of authentication of the Prover to the Issuer: any authentication protocol can be used to enable the Issuer to ascertain himself of the identity of the Prover. The Issuer then looks up the attributes he wishes to encode in the token, and agrees upon the values of these attributes with the Prover. Next, the Prover chooses the contents of the Prover information (PI) field, as well as the private key α^{-1} with $\alpha \in_R \mathbb{Z}_q^*$ of the token. Furthermore, he chooses the values $\beta_1, \beta_2 \in \mathbb{Z}_q$, which he uses to obfuscate the contents of the message which he presents to the Issuer for signing. The issuer, on the other hand, generates $w \in_R \mathbb{Z}_q$. After this setup phase, they engage in a three-step message exchange which results in the Prover possessing all the contents of the token as described in Subsection 4.3, including some parts which the Issuer has not seen, namely the public key h, the Prover information field PI and the three parts of the Issuer (blind) signature σ'_Z, σ'_c and σ'_r. Furthermore, he has the private key α^{-1} of the token. The message flow of the protocol is depicted in Figure 4.5.

4.6 Presenting a token

The presentation protocol is used by the Prover and the Verifier to present the values of certain attributes of a token. This protocol is non-interactive: to prove that he possesses a token which contains certain attributes, the Prover does some calculations to generate a presentation proof. He sends this proof, along with the public parts of the token, to the Verifier. The Verifier subsequently verifies that the proof is indeed valid, i.e. that the proof indeed shows that the Prover possesses a token which contains the agreed upon attributes. This protocol is the non-interactive version of a zero-knowledge protocol. In the

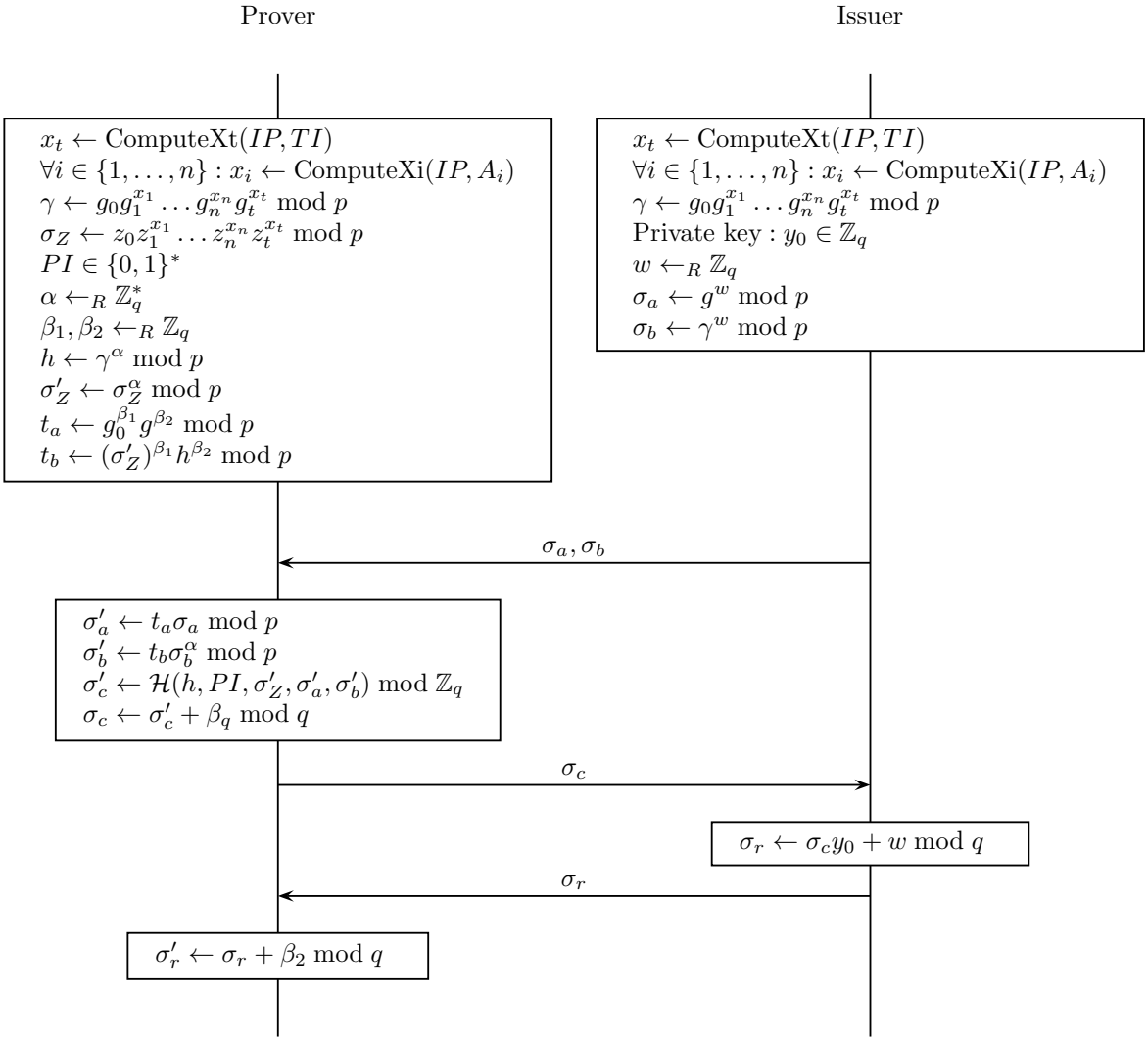

Fig. 4.3: Blind signing protocol which yields a signature on the U-Prove token.

intermezzo, we explained the three-step nature of most zero-knowledge protocols: First, the Prover commits to a certain (random) value, then the Verifier picks a challenge, upon which the Prover generates the appropriate response. In the case of U-Prove proofs, an other approach is taken: First, the Prover performs *all* calculation steps, then sends a transcript of the generated message sequence to the Verifier. Of course, the validity of a zero-knowledge protocol hinges on the inability of the Prover to choose the challenge himself. Therefore, the challenge is generated using some deterministic algorithm (often, a hash function) on the commitment and other input values. The Verifier can repeat this process to ascertain himself of the fact that the Prover actually possesses the private key of the token. This technique is more broadly known as the Fiat-Shamir heuristic [8].

Here, we describe two types of presentation proofs. One, the regular presentation proof, is described fully in [10] – we summarize it here. The other, the interval presentation proof, is our own adaption of an interval proof which was presented in [18] and of Okamoto's identification protocol as described in [15], which is covered at length in Appendix A.

Regular presentation proofs

A Prover uses a regular presentation proof to show a subset of the attributes which are encoded in the token. The indexes of the attributes which he wishes to disclose are the set $D \subseteq \{1, \ldots, n\}$, and we denote the set of undisclosed indices with $U = \{1, \ldots, n\} \setminus D$. We assume that the Prover has full access to the token and all attributes. To generate the proof, the Prover proceeds as described in Algorithm 1.

Verification In order to verify the presentation proof which was presented by a Prover, the Verifier performs Algorithm 2 (we assume that the Verifier has access to all public values of the token, as well as the presentation proof which was sent by the Prover). Note that the Verifier does not require the values of the attributes which are not disclosed to verify the proof.

Interval presentation proofs

The current version of the U-Prove specification does not allow the construction of proofs which make statements about properties of attributes [10]. Microsoft has indicated that this may be a feature in a subsequent version of the specification, but they have not given a specific timeframe in which this might be expected. Here, we present a limited extension of the specification in this direction: using our interval presentation proof, the Prover can disclose one specific piece of information: that the value of one particular attribute is contained in a certain interval. This is specifically useful in the case of numerical attributes: a date of birth which is more than 18 but less than 65 years ago, or a school grade which is high enough to pass.

We assume that the Prover has a token with public key h which encodes n attributes $A_1, \ldots, A_n$ into their internal representation $x_1, \ldots, x_n$. We name the private key of the token α. Assume that the Prover wishes to show that

Algorithm 1 Presentation proof

Require: Issuer parameters IP, indices of attributes disclosed D, undisclosed U, message $m \in \{0,1\}^*$, token $\mathcal{T}$, private key $\alpha^{-1} \in \mathbb{Z}_q$, attribute values $(A_1, \ldots, A_n) \in \{0,1\}^*$

Ensure: Output a proof which could only have been generated by a Prover who has knowledge of some $\alpha, (x_1, \ldots, x_n, x_t)$ such that $h = (g_0 g_1^{x_1} \ldots g_n^{x_n} g_t^{x_t})^{\alpha}$ and which requires only the disclosure of $\{A_i\}_{i \in D}$ to allow verification.

$x_t \leftarrow \text{CalculateXt}(IP, TI)$
for $i = 1 \rightarrow n$ **do**
  $x_i \leftarrow \text{CalculateXi}(A_i)$
end for
for all $i \in U \cup \{0\}$ **do**
  $w_i \leftarrow_R \mathbb{Z}_q$
end for
$a \leftarrow \mathcal{H}(h^{w_0} \prod_{i \in U} g_i^{w_i} \bmod p)$
$c \leftarrow \text{GenerateChallenge}(IP, \mathcal{T}, a, m, D, \{x_i\}_{i \in D})$
$r_0 \leftarrow c\alpha^{-1} + w_0 \bmod q$
for all $i \in U$ **do**
  $r_i \leftarrow -cx_i + w_i \bmod q$
end for
return $\{A_i\}_{i \in D}, a, r_0, \{r_i\}_{i \in U}$, Token public key

Algorithm 2 Verify Presentation Proof

if not VerifySignature(Token public key) **then**
  return false
end if
$x_t \leftarrow \text{CalculateXt}(IP, TI)$
for $i \in D$ **do**
  $x_i \leftarrow \text{CalculateXi}(A_i)$
end for
$c \leftarrow \text{GenerateChallenge}(IP, \mathcal{T}, a, m, D, \{x_i\}_{i \in D})$
if not $a = \mathcal{H}((g_0 g_t^{x_t} \prod_{i \in D} g_i^{x_i})^{-c} h^{r_0} \prod_{i \in U} g_i^{r_i} \bmod p)$ **then**
  return false
end if
return true

$x_j \in [a, b)$ (with $1 \leq j \leq n$). To achieve that, the Prover runs Algorithm 3. Algorithms 4 and 5 are the subalgorithms called from Algorithm 3. Algorithm 5 is the non-interactive, multiple variable version of the Okamoto Identification Protocol. We elaborate on the derivation of these constructions in Appendix A.

Algorithm 3 IntervalProof

Require: $p, q, n, (x_1, \ldots, x_n), x_t, (g_0, \ldots, g_n, g_t), \alpha, j, a, b,$ message m

Ensure: Output a proof which could only have been generated by a Prover who has knowledge of some $\alpha, (x_1, \ldots, x_n, x_t)$ with $x_j \in [a, b)$ such that $B = (g_0 g_1^{x_1} \ldots g_n^{x_n} g_t^{x_t})^{\alpha}$ and which does not yield any additional information to the Verifier.

for all $i \in \{1, \ldots, n, t\}$ **do**
$g_i' \leftarrow \frac{1}{g_i}$
end for
$\alpha' \leftarrow \frac{1}{\alpha}$
$\alpha_1' \leftarrow_R \mathbb{Z}_q$
$\alpha_2' \leftarrow \alpha' - \alpha_1'$
$g_0^{(2)} \leftarrow h^{\alpha_2'} g_j'^{x_j}$
$g_0^{(1)} \leftarrow g_0 (g_0^{(2)})^{-1}$
$\mathcal{O} \leftarrow$ OkamotoProof$(p, q, n{+}1, (\alpha_1', x_{1,\ldots,j-1,j+1,\ldots,n,t}), (h, g'_{1,\ldots,j-1,j+1,\ldots,n,t}), m)$
$\mathcal{B}_1 \leftarrow$ BasicIntervalProof$(p, q, x_j - a, h, g_j', \alpha_2', k, m)$
$\mathcal{B}_2 \leftarrow$ BasicIntervalProof$(p, q, x_j - b + 2^k, h, g_j', \alpha_2', k, m)$
return $\mathcal{O}, \mathcal{B}_1, \mathcal{B}_2$

Algorithm 4 BasicIntervalProof

Require: $p, q, x, g, h, \alpha, k, m$

Ensure: Output a proof which could only have been generated by a Prover has knowledge of some α, x with $x \in [0, 2^k)$ such that $g^\alpha h^x = B$ and which does not yield any additional information to the Verifier.

$B \leftarrow g^\alpha h^x$
for $i = 0 \rightarrow k-1$ **do**
  $x_i \leftarrow_R \{0,1\}$ such that $\sum_{i=0}^{k-1} x_i 2^i = x$
  $\alpha_i \leftarrow_R \mathbb{Z}_q$ such that $\sum_{i=0}^{k-1} \alpha_i 2^i = \alpha$
  $B_i \leftarrow g^{\alpha_i} h^{x_i}$
  $w_i, r_{i,1-x_i}, d_{i,1-x_i} \leftarrow_R \mathbb{Z}_q$
  $a_{i,1-x_i} \leftarrow g^{r_{i,1-x_i} + \alpha_i d_{i,1-x_i}} h^{(-1)^{1-x_i} d_{i,1-x_i}}$
  $a_{i,x_i} \leftarrow g^{w_i}$
end for
$c \leftarrow \mathcal{H}(B, m, \alpha_{i,j})$, with $i \in \{0, \ldots, n\}$ and $j \in \{0,1\}$
for $i = 0 \rightarrow k-1$ **do**
  $d_{i,x_i} \leftarrow c - d_{i,1-x_i}$
  $r_{i,x_i} \leftarrow w_i - \alpha_i d_{i,x_i}$
end for
return $a_{i,j}, r_{i,j}, d_{i,j}, B_i$, with $i \in \{0, \ldots, k-1\}$ and $j \in \{0,1\}$

Algorithm 5 OkamotoProof

Require: $p, q, n, (g_1, \ldots, g_n), (x_1, \ldots, x_n), m$

Ensure: Output a proof which could only have been generated by a Prover who has knowledge of some $x_1, \ldots, x_n$ such that $\prod_{i=1}^n g_i^{x_i} = h$ and which does not yield any additional information to the Verifier.

$h \leftarrow \prod_{i=1}^n g_i^{x_i}$
for $i = 1 \rightarrow n$ **do**
  $u_i \leftarrow_R \mathbb{Z}_q$
end for
$a \leftarrow \prod_{i=1}^n g_i^{u_i}$
$c \leftarrow \mathcal{H}(h, m, a)$
for $i = 1 \rightarrow n$ **do**
  $r_i \leftarrow u_i + c x_i$
end for
return $a, r_1, \ldots, r_n$

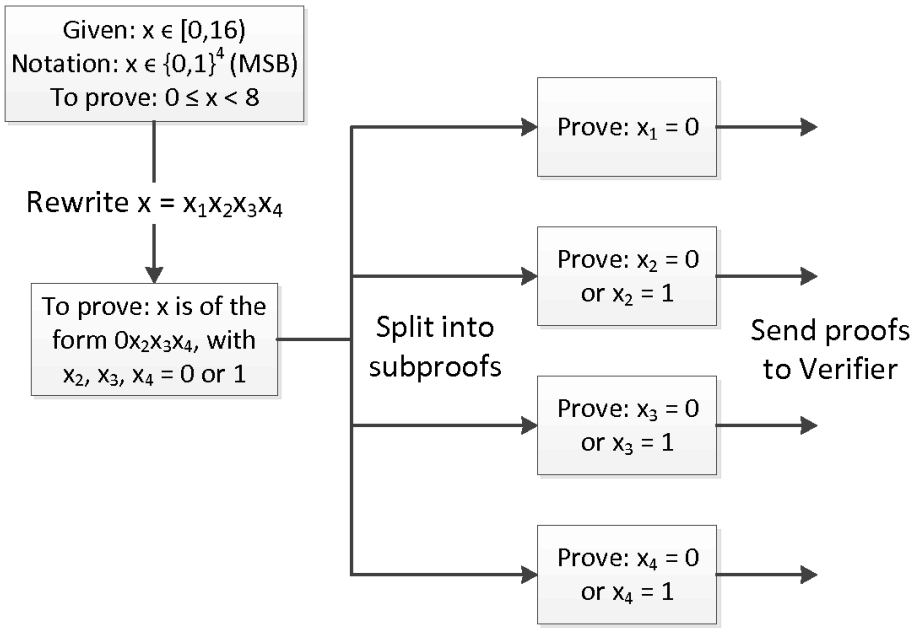

Fig. 4.4: Outline of the steps taken to produce a basic interval proof, as performed in Algorithm 4. In this example, the Prover proves that the value of the attribute to disclose, $x \in [0, 16)$, is contained in the interval $[0, 8)$.

Verification In order to verify an interval proof which was generated with the above algorithm, the Verifier would verify the output of the three calls to the subalgorithms (Algorithms 4 and 5). The required calls to the respective verification procedures are described in Algorithm 6. For the convenience of the reader, we present correctness proofs of the basic interval proof and the Okamoto proof and their respective verification algorithms here.

In order to prove correctness of the Okamoto proof, we take the verification algorithm and show that, given the steps in the proof generation algorithm, the check which is performed there always succeeds. The check which is performed there is

$$\prod_{i=1}^{n} g_i^{r_i} \stackrel{?}{=} ah^c.$$

Now, we expand ah^c to show that it is, indeed, equal to the left side of the expression. We use assignments from Algorithm 5:

$$\begin{aligned} ah^c &= \left(\prod_{i=1}^{n} g_i^{u_i}\right)\left(\prod_{i=1}^{n} g_i^{x_i}\right)^c \\ &= \left(\prod_{i=1}^{n} g_i^{u_i+cx_i}\right) \\ &= \left(\prod_{i=1}^{n} g_i^{r_i}\right). \end{aligned}$$

We use the same procedure to prove correctness of the basic interval proof. We use the assignments from Algorithm 3. In the verification algorithm, there are three checks which are performed for each $i \in \{0, \ldots, k-1\}$. Let, therefore, such an i be given. We reason as follows:

- By definition, it holds that $d_{i,x_i} + d_{i,1-x_i} = c$. Therefore, the first step of the verification succeeds.
- If $x_i = 0$, we have that

$$\begin{aligned} a_{i,0} &= g^{w_i} \\ &= g^{w_i-\alpha_i d_{i,0}} g^{\alpha_i d_{i,0}} \\ &= g^{r_{i,0}} g^{\alpha_i d_{i,0}} \\ &= g^{r_{i,0}} (B_i)^{d_{i,0}} \end{aligned}$$

  which satisfies the second step of the verification. Also, we have that

$$\begin{aligned} a_{i,1} &= g^{r_{i,1}+\alpha_i d_{i,1}} h^{-d_{i,1}} \\ &= g^{r_{i,1}} g^{\alpha_i d_{i,1}} h^{-d_{i,1}} \\ &= g^{r_{i,1}} (B_i)^{d_{i,1}} (h^{-1})^{d_{i,1}} \\ &= g^{r_{i,1}} (B_i/h)^{d_{i,1}}, \end{aligned}$$

  which satisfies the third step of the verification.

- If $x_i = 1$, we have that

$$\begin{aligned} a_{i,0} &= g^{r_{i,0}+\alpha_i d_{i,0}} h^{d_{i,0}} \\ &= g^{r_{i,0}} g^{\alpha_i d_{i,0}} h^{d_{i,0}} \\ &= g^{r_{i,0}} (B_i)^{d_{i,0}}, \end{aligned}$$

  which satisfies the second step of the verification. Also, we have that

$$\begin{aligned} a_{i,1} &= g^{w_i} \\ &= g^{r_{i,1}} g^{\alpha_i d_{i,1}} \\ &= g^{r_{i,1}} g^{\alpha_i d_{i,1}} (hh^{-1})^{d_{i,1}} \\ &= g^{r_{i,1}} (g^{\alpha_i} h)^{d_{i,1}} (h^{-1})^{d_{i,1}} \\ &= g^{r_{i,1}} (B_i)^{d_{i,1}} (h^{-1})^{d_{i,1}} \\ &= g^{r_{i,1}} (B_i/h)^{d_{i,1}}, \end{aligned}$$

  which satisfies the third step of the verification.

We conclude that all three steps of the verification have been satisfied for every $i \in \{0, \ldots, k-1\}$. Now, we show that the subcommitments B_i indeed combine to form the original commitment B:

$$\begin{aligned} \prod_{i=0}^{k-1} (B_i)^{2^i} &= \prod_{i=0}^{k-1} (g^{a_i} h^{x_i})^{2^i} \\ &= \left(\prod_{i=0}^{k-1} (g^{a_i})^{2^i} \right) \left(\prod_{i=0}^{k-1} (g^{a_i})^{2^i} \right) \\ &= g^{\sum_{i=0}^{k-1} \alpha_i 2^i} h^{\sum_{i=0}^{k-1} x_i 2^i} \\ &= g^{\alpha} h^{x}, \end{aligned}$$

which satisfies the last verification requirement.

Algorithm 6 VerifyIntervalProof

Require: $p, q, n, (g_0, \ldots, g_n, g_t), g_0^{(1)}, g_0^{(2)}, j, a, b, \mathcal{B}_1, \mathcal{B}_2, \mathcal{O}, m$.
Ensure: Output true iff the interval proof holds, i.e. it convinces the Verifier that the Prover indeed has knowledge of all attributes, and that $x_j \in [a, b)$.

for all $i \in \{1, \ldots, n, t\}$ **do**
  $g_i' \leftarrow \frac{1}{g_i}$
end for
if $g_0^{(1)} g_0^{(2)} \neq g_0$ **then**
  return false
end if
if not VerifyOkamotoProof$(p, q, n+1, (h, g'_{1,\ldots,j-1,j+1,\ldots,n,t}), g_0^{(1)}, \mathcal{O}, m)$ **then**
  return false
end if
if not VerifyBasicIntervalProof$(p, q, h, g_j', k, g_0^{(2)} g_j'^{-a}, \mathcal{B}_1, m)$ **then**
  return false
end if
if not VerifyBasicIntervalProof$(p, q, h, g_j', k, g_0^{(2)} g_j'^{-b+2^k}, \mathcal{B}_2, m)$ **then**
  return false
end if
return true

Algorithm 7 VerifyOkamotoProof

Require: $p, q, n, (g_1, \ldots, g_n), h, \mathcal{O} = (a, r_1, \ldots, r_n), m$
Ensure: Output true iff the Okamoto proof holds, i.e. it convinces the Verifier that the Prover indeed has knowledge of some $x_1, \ldots, x_n$ such that $\prod_{i=1}^n g_i^{x_i} = h$.

$c \leftarrow \mathcal{H}(h, m, a)$
if $\prod_{i=1}^n g_i^{r_i} \neq a h^c$ **then**
  return false
end if
return true

Algorithm 8 VerifyBasicIntervalProof

Require: $p, q, g, h, k, B, \mathcal{B} = (a_{i,j}, r_{i,j}, d_{i,j}, B_i), m$, with $i \in \{0, \ldots, k-1\}$ and $j \in \{0, 1\}$.

Ensure: Output true iff the basic interval proof holds, i.e. it convinces the Verifier that the Prover indeed has knowledge of some α, x with $x \in [0, 2^k)$ such that $g^\alpha h^x = B$.

$c \leftarrow \mathcal{H}(B, m, \alpha_{i,j})$
for all $i \in \{0, \ldots, k-1\}$ **do**
  if $c \neq d_{i,0} + d_{i,1}$ **then**
    return false
  end if
  if $a_{i,0} \neq g^{r_{i,0}}(B_i)^{d_{i,0}}$ **then**
    return false
  end if
  if $a_{i,1} \neq g^{r_{i,1}}(B_i/h)^{d_{i,1}}$ **then**
    return false
  end if
end for
if $B \neq \prod_{i=0}^{k-1}(B_i)^{2^i}$ **then**
  return false
end if
return true

5. CASE STUDY: OV-CHIPKAART

> The soul of the journey is liberty, perfect liberty, to think, feel, do just as one pleases.
>
> *On Going a Journey*, William Hazlitt

In this chapter, we will perform a case study of the OV-chipkaart, the Dutch contactless smartcard which is used for almost all transactions in the public transport. We will first describe the system which is currently in place, as well as the relevant aspects of the underlying architecture. Next, we will describe the points of view of various stakeholders in the debate surrounding the privacy aspects of the OV-chipcard. To conclude, we formulate our own view upon the problems which may be encountered when trying to actually implement U-Prove in the OV-chipkaart system.

5.1 Environment

The public transport market in the Netherlands is highly fragmented: many forms of public transport are offered by multiple (usually regional) companies. In 2001 [21], the five largest public transport companies founded the joint venture Trans Link Systems (TLS). The goal of TLS was to produce, sell and exploit a national card which would be used for payment in all public transport transactions: the OV-chipkaart[1]. The first cards were issued in April 2005, and after that, the OV-chipkaart has steadily become ubiquitous. As of July 2011, the OV-chipkaart is the only valid ticket for regional public transport in eight out of twelve provinces of the Netherlands[2]. Only very few regional train services remain which do not support the OV-chipkaart yet.

The introduction of the OV-chipkaart has led to considerable controversy. First, German hackers successfully hacked the chipcard which is used for the OV-chipkaart, the Mifare Classic. Next, researchers of the Radboud University Nijmegen were able to leverage this possibility to access and modify the information which is stored in an OV-chipkaart. Furthermore, the Dutch regulatory office for data protection (CBP) has ruled that the public transport companies did not observe the legally required procedures in their handling of personal data of their customers [16]. This has urged the digital civil rights group Bits of Freedom to award Trans Link Systems two Big Brother Awards in 2010 [3].

[1] 'OV' stands for 'Openbaar Vervoer', which is Dutch for 'Public Transport'.

[2] `http://www.ov-chipkaart.nl/allesoverdeov-chipkaart/waarreizen/waartegebruiken/`

[3] `https://www.bigbrotherawards.nl/wp-content/uploads/2010/01/Persbericht-Winnaars-Big-Brother-Awards-bekend1.pdf`

In the light of these events, TLS and several partners have established Open Ticketing, a foundation which aims at sharing the knowledge of e-ticketing which is acquired through the OV-chipkaart project. Additionally, TLS and Open Ticketing cooperate to design the follow-up technology of the OV-chipkaart. This technology should be both resistant to hacks such as the ones performed by the aforementioned researchers and, ideally, maximize traveller privacy by storing only a minimal amount of information in centralized databases. U-Prove could play a role in the cryptographic protocols which achieve these goals.

Methodology In this chapter, we present views of multiple involved parties with regard to the possibility of adopting U-Prove for the OV-chipkaart system. It should be noted that these views were formulated on the basis of interviews with multiple involved parties. While we have strived to accurately represent the opinions voiced during the interviews, it is possible that we misinterpreted some of these opinions.

The view of Open Ticketing was derived from an interview with Bas van Os, director of Open Ticketing. As Open Ticketing cooperates extensively with the academic world, he was already aware of U-Prove and the possibilities it offers. Therefore, the view of Open Ticketing is much more detailed than the others. The view of the public transport companies was extrapolated from the view of the Dutch national railway company NS. This was derived from an interview with Sergej Katus, privacy officer of NS, and presentations by Sergej Katus and Pim Bonenkamp, security officer of NS, at the EurPrivacy symposium in Nijmegen on March 24, 2011. The view of Trans Link Systems was derived from documents on their website, public and press statements. The interviews with Sergej Katus and Bas van Os were instrumental in our interpretation of this view, as their cooperation with TLS yielded an otherwise unaccessible perspective on the view of TLS, as TLS itself proved unresponsive to our requests for further information (i.e. other than that which is published on the website) with regard to their approach to protecting traveller privacy. Since the public transport traveller organization ROVER does not seem to have a strong opinion with regard to traveller privacy and the OV-chipcard, we have represented the interests of the travellers in the interests of Open Ticketing (with regard to traveller privacy) and the public transport companies (with regard to functionality and ease of use).

5.2 Existing structures

In order to effectively reflect upon the difficulties and opportunities which arise when applying U-Prove to the context of the OV-chipkaart, it is important to describe the organisational structure of the system [24]. This is most easily studied when looking from the perspective of the life cycle of an OV-chipkaart as it it currently used.

Issuance An OV-chipkaart is issued by Trans Link Systems or one of its partners. There are three different types of cards [23], two of which are for long term use: a peronalised and an anonymous variant. The third, a disposable card, has a different internal architecture from the other two, and it can only be used to

store a single travel product on the card. The disposable card does not allow purchase of multiple travel products, and it does not contain a balance. Upon issuance, a unique 16-digit identification number is assigned to and stored on the card. This number is also stored in a central database which is managed by Trans Link Systems and linked to the account of the card holder. For personalised cards, additional credentials are stored in the central database, as well as on the card [24]. On the card, several different types of information are stored. On the outside of the personalised card, the name, date of birth and photograph of the owner are diplayed, along with the 16-digit identification number. In the chip, some more operational information is stored, such as the type of card, any travel products the owner has purchased and the current balance. Only two identifying credentials are stored on the card: the date of birth of the owner, and the unique 16-digit identification number. Note that, originally, the public transport system in the Netherlands used an attribute-based payment system, where a traveller buys a specific travel product and shows this to display that he is permitted to travel to his destination. The OV-chipkaart is, however, an identity-based payment system, which links all purchases and travel movements of a traveller to his identity. As U-Prove is attribute-based as well, using U-Prove will probably yield a travel payment system which is closer to the original (anonymous) system in its privacy guarantees.

Purchasing travel products The OV-chipkaart can be used as a digital wallet to store money which can be used to perform the transactions necessary for normal travel. A customer can receive additional products which entitle him to a discount, free travelling or other perks. These products can be issued by any participating public transport company and are registered both in the back-end database of TLS and stored on the card of the customer. Furthermore, a customer can increase his balance by paying the appropriate amount at a charging terminal or at the counter of a public transport company. He can also authorize TLS to automatically deduct a fixed amount from his account when the balance on his OV-chipkaart drops below a certain threshold.

Travelling In order to use his OV-chipkaart to travel using public transport, a customer has to check in at a terminal in a station or vehicle. Such a terminal contains an RFID reader to communicate with the card. When the customer checks in, his card will first identify itself to the terminal, after which the terminal will determine whether any products are stored on the card which are relevant to the current trip. Using this information, the terminal deducts an amount from the balance on the card, which will almost always be more than the trip will cost. A registration of this check-in is made on the card. If a conductor wishes to verify whether a traveller is indeed entitled to travel on that vehicle, he will use his off-line mobile RFID reader which verifies that the card is currently in the checked in state. This interaction is logged on the device, and it is synchronized with the back-office of TLS at the end of the day. When the traveller reaches his destination, he again holds his card against the terminal, which will determine the actual amount which the trip will cost. It uses the information about the previously deducted amount together with the final amount to determine whether to restore some of the deducted balance or to deduct even more. After this last transaction, the balance on the card represents the actual

balance to which the customer is entitled. On the other hand, since most terminals are off-line, the back-office is not yet aware of the transactions which have occurred during the day. At the end of every day, all terminals are connected to the back-office in order to exchange information about the transactions which have occurred.

Revocation/expiration/fraud detection Each OV-chipkaart has an (internally stored) expiration date, after which the customer should no longer be able to use the card. Furthermore, the recent attacks against the protocols and architecture of the OV-chipkaart have led to a stronger need to effectively detect and prosecute OV-chipkaart fraud. Conductors are, as always, tasked to find travellers who do not possess a valid travel document. Tampering with the internals of the OV-chipkaart is mostly detected in the backoffice systems, which probably[4] results in addition of the card identification number to a card revocation list which is sent daily to the terminals. In the case of personal cards, the identity of the fraudster can easily be retrieved from the back-office database. In the case of anonymous cards, this is not as easily achieved. It is speculated that TLS or its partners might employ secondary channels to find and identify fraudsters. For example, if a credit or debit card is used to increase the balance of the OV-chipkaart, then, given access to the appropriate databases, it is possible to link the account number to the OV-chipkaart. On the other hand, TLS has stated [22] that it will not secretly attempt to discover the identities of holders of anonymous cards. As OV-chipkaart fraud constitutes a felony [1], investigative services might employ this technique. While it is clear that they are authorized to do so, it is unkown whether this technique is actively used.

5.3 Involved parties, requirements and proposed solutions

The current implementation of the OV-chipkaart does not promote privacy-friendly handling of the personal information of travellers, as the privacy enhancing technologies that are applied are revocable without the cooperation of the traveller, e.g. for investigative purposes. In this debate, multiple stakeholders each have different views of the matter, with different opinions forming based on their own demands and interests. We will describe the vision of each of these stakeholders separately.

Trans Link Systems TLS was started with a single objective: to eventually replace all tickets used in the Dutch public transport infrastructure by one system, a contactless smartcard. This will enable public transport companies to effectively distribute the revenues they generate. In contrast, the old system called 'strippenkaart' did allow infrequent customers to use one (non-smart) card to pay for almost all public transport[5], but since it was not digital, it was impossible to track which public transport company these travellers used. Furthermore, since subscription card holders also did not check in with their subscription cards, the size of this volume of customers could also not be assessed. Moreover, since OV-chipkaart fraud has become a fairly regular occurence, effective fraud detection and prosecution is an absolute necessity for TLS. TLS

[4] TLS is not very forthcoming with details on their methods of fraud detection.
[5] The strippenkaart was not valid in most trains.

is conscious of the legal requirements of the OV-chipkaart and uses these to benchmark whether the current implementation adheres to these requirements [26, 3]. Even though TLS has published a document about their adherence to these regulations[6], they are perceived as a privacy-invading party by the general public[7]. The point of view of Trans Link Systems is that, even though the encryption on the card has been broken, no immediate measures have to be taken, as the risks resulting from this vulnerability are solely for the public transport companies[8]. They have announced a new version of the OV-chipcard which will contain an RFID chip which is harder to break. This version will be introduced in the next few years[9]. On the other hand, no functional differences with regard to privacy have been announced. This seems to stem from a set of three ideas with regard to privacy protection:

- The current architecture is already compliant with Dutch legislature, in particular the Data Protection Act. Going any further would only muddle the debate: the privacy advocates will always be asking for more privacy protection.
- Any limitations of the functionality which are introduced in the light of further privacy protection can immediately lead to loss of revenue and increased operational costs.
- As the need for further privacy protection is not broadly felt across the population, the expected profits from an improved public perception of Trans Link System's approach to privacy are low.

Public Transport Companies The public transport companies have repeatedly been criticized on the way they handle customer data. When a customer purchases a product from such a company, the company receives the personal information about this person from TLS, as it is subsequently allowed to contact this person with relevant travel details. Also, the Dutch telecommunication act [2] allows these companies to directly market related products to these customers. Other than that, the companies are allowed insight into the personal details of customers only under strict conditions, such as for restitution purposes in case of extensive delays. Pseudonymized data can be used for research purposes, which gives these companies a significant advantage over would-be competitors which intend to operate on the same trajectories, as they have more intricate knowledge about the precise travel movements of their customers. In the privacy debate surrounding the OV-chipkaart, it is TLS who is portrayed as responsible for the decisions with respect to privacy and the architecture of the system. This means that, even though the reasoning of TLS applies to the public transport companies as well, their incentive to change the public's perception of the privacy aspects of the OV-chipkaart is even lower. The view of the public transport companies with regard to the possibility of incorporating U-Prove in the OV-chipkaart system is explained in the next paragraph. We expect that this

[6] http://www.ov-chipkaart.nl/pdf/22246/privacyovchipkaart

[7] https://www.bigbrotherawards.nl/wp-content/uploads/2010/01/Persbericht-Winnaars-Big-Brother-Awards-bekend1.pdf

[8] http://www.ov-chipkaart.nl/pdf/speech

[9] http://www.volkskrant.nl/vk/nl/2680/Economie/article/detail/1828810/2011/01/27/Translink-Systems-nog-geen-nieuwe-ov-chip-gekozen.dhtml

view will also be held by TLS, as the interests and requirements of the two are similar.

As with every use case of U-Prove, it is important to first define which parties play the role of the Issuer, the Prover and the Verifier. The privacy officer of NS, the Dutch railroad company, put forward the following distribution. Every traveller possesses a contactless smartcard which can be personalised or anonymous. Both variants contain a unique identifier as an attribute. The personalised variant additionally contains some more attributes, which encode the personal information of the owner which is currently also stored in the OV-chipkaart. This information is stored in a token which is issued by the Issuer, which is any one of the public transport companies or TLS. Additionally, a customer may purchase travel products from a public transport company. In order to do this, he uses the token which is stored on his smartcard to disclose his unique identifier to the public transport company. This company then issues a token which contains attributes which describe the purchased travel product and the unique identifier of the owner of the card. This token is also stored on the smartcard. Now, when the owner of the smartcard is asked to present his card at a gate in a station or to a conductor, the smartcard plays the role of the Prover, by performing the presentation protocol with respect to the information relevant to the entry moment: the unique identifier of the owner and the relevant information about any travel products which are stored on the card. This information can be minimalised (e.g. only prove that you are entitled to the discount, without disclosing the specific product you use), but the full information about the traveller is always stored in the database of TLS. By linking this information through the unique identifier, it is possible to check the entire transaction afterwards. This enables TLS and the public transport companies to effectively detect and combat fraud: if any irregularity arises, the unique identifier can be blocked and registered in an identifier revocation list which is subsequently passed on to the terminals of gates and conductors. In this vision therefore the privacy of the traveller is protected against malicious interference at the conductor or gate terminal level. However, at the backoffice, the privacy of the traveller is approached in the same way as it currently is. This system does not facilitate updates of issued tokens: the balance which is stored on the OV-chipkaart is not encoded in any token.

Open Ticketing After the launch of the OV-chipkaart project, it became apparent that there was a need to actively share the lessons learned by TLS and the public transport companies when introducing the card. In other countries, similar systems are considered or are already in use. These could possible benefit from the body of knowledge that was built in the Netherlands. Furthermore, there was a political need to reconsider the underlying architecture of the OV-chipkaart system. To achieve these goals, the Open Ticketing foundation was established. This foundation, which cooperates closely with TLS while being formally independent from it, promotes the active sharing of the knowledge acquired while introducing the OV-chipkaart. It also functions as a point of coordination of the activities to design an improved architecture for the next generation of the OV-chipkaart. With regard to the possibilities of incorporating U-Prove in the OV-chipkaart system, the view of the Open Ticketing

foundation is as follows.

Each traveller has a smartcard, which is issued by a central organisation like TLS. However, this card contains no identifying information which is presented at any transaction. Instead, multiple Issuers each contribute a token to the smartcard. The combination of these tokens enables Verifiers to infer whether a traveller should be allowed to pass a gate or complete a conductor check. An important principle in this vision is that, a priori, the smartcard platform is trusted to not allow any unregulatory modifications. Therefore, the fraud detection occurs in a way which differs essentially from the current approach or the one in the vision of the public transport companies. Also, in contrast to the current approach, this approach is not necessarily universal: users could be given the choice whether they want the 'normal' or the privacy-enhanced OV-chipkaart. Furthermore, the vision which Open Ticketing is currently developing into a specification for a viable next-generation OV-chipkaart is open, which means that anybody can read the specification and try to implement his own similar system. Public transport companies from many other countries can therefore cooperate on further development of this technology. Also, any existing weaknesses in the specification will be more easily found by the (academic) community.

Three different Issuers are responsible for a token on the smartcard. First, the card issuer issues a token which contains any attributes which may be important for subsequent travelling transactions, such as the owner's date of birth. The card issuer can be TLS, but that is not necessary. It is important that the card issuer is considered to be authorative with respect to the attributes for which he issues the token. Second, the owner of the smartcard may purchase a specific travel product at a public transport company. In such a case, the public transport company issues a token which contains attributes about this travel product, along with the public key of the token which contains personal information. Note that this is essentially different from creating a persistent link with the owner of the product, as the token with personal information is both card-specific and non-persistent. In addition to this transaction, the public transport company may register the traveller in their own database, but as that registration will contain no data from the OV-chipkaart domain, that is outside our scope. Note that such a travel product may also take the form of an increase in the card balance: A monetary unit of a certain fixed value is placed on the card, along with a token which contains the amount stored on the card as an attribute, along with a non-unique revocation identifier. Third, when checking in at a gate, the terminal issues a token which contains the time and point of entry to the public transport system as attributes. When checking out, the terminal checks the distance of the exit gate to the entrance gate (which is a derived attribute of the point of entry)[10]. It then determines how much the trip costs from this distance, along with any other information it receives from the card in the form of presentation of the other tokens it contains.

[10] It is disputed whether this feature is a realistic requirement of a privacy-enhanced OV-chipkaart.

This system allows for some fraud detection, as there is partial information stored with the balance tokens. Concretely, this means that some non-unique revocation identifier will be stored in each such token, preferably in a token information field (which is always disclosed). Now, if fraud is detected with a card, this identifier can be retrieved from the presentation proof, and used to determine in which 'batch' of balance tokens the fraudulent token was issued. By adjusting the size of these batches, the accuracy of fraud detection (and, reversely, the anonimity of the user) can be modified to any given value - Open Ticketing suggests batches of 20,000 euros. If fraud is detected with any balance token from a batch, all tokens from that batch will be revoked. Now, all owners of such a balance token can exchange their token for a new one by proving that their card was not involved in the fraudulent transactions. This examination could, for example, be done at the counter of a public transport company. This procedure would somewhat increase the load on non-fraudulent passengers (as they would have to 'unlock' their balance if someone performs fraud with a balance token from the same batch). However, in such a setting, there is always a tradeoff between user convenience and user anonimity, as the traceability of a transaction to a user automatically implies this user is no longer anonymous.

The attributes which the different Issuers store on the OV-chipkaart are not updated after issuance, except when they are replaced by another token from the same Issuer. In the case of the balance token, this means that the OV-chipkaart is trusted to perform these transactions faithfully. Should it not do so, the fraud detection procedure which was described earlier is applied.

Requirement	Current solution	Vision of PTCs	Vision of Open Ticketing
TLS and PTCs			
Analysis of traveller data is possible and legal	Analysis of the data is legal	Analysis of the data is legal	All data is anonymized, but analysis is still possible. Fewer security measures are necessary.
Effective fraud detection is possible	This is the baseline for fraud detection possibilities	All fraud current fraud detection possibilities still exist	Fraud detection is done by grouping users in balance increase pools.
Solution is compliant with legislation	CBP has critized PTCs for their data storage	Same data is stored centrally as is currently the case.	No identifying data is stored centrally
Solution is viable in day-to-day use	Solution is already in use	Solution uses no resource-intensive U-Prove functionality (e.g. derived attributes)	Solution might require many presentation proofs at a gate or conductor, long check-in times.
Open Ticketing			
Solution collects minimal amount of personal data required	Solution only complies with absolute minimum requirements (i.e. legal)	Less data is collected at terminals, but just as much centrally as is the case now	Collecting less data would make core functionality unworkable
Specification of solution is open	Using a closed, non-public crypto algorithm has resulted in hack of the card	This is not specified	This specification is open
Solution requires minimal amount of end-user trust	User trusts implementation on card, terminals and backoffice	User trusts implementation on card and backoffice	User trusts implementation on card

Tab. 5.1: Summary of the extent to which the proposed solutions fulfill the requirements of the involved parties (green: Fully compliant, yellow: Partially compliant, red: Not compliant).

5.4 Conflicts

As the requirements and the vision of the two primary stakeholders are compared, it becomes apparent that the differences between the two can not be resolved by merely compromising on the value of some parameters: their views of the OV-chipkaart and its privacy requirements are fundamentally different. We discuss several of these differences here.

Even though TLS was awarded a Big Brother Award in 2011, their perception of the OV-chipkaart system with regard to privacy is quite positive. The public transport companies share this view of the matter: when comparing the protection measures that have been implemented in the OV-chipkaart system with the requirements stated in the Dutch Data Protection Act, they see no reason to change their technical measures or processes. On the other hand, the Dutch Data Protection Authority has announced that the storage of traveller data has significant shortcomings in the light of this Act: too much data is stored by the public transport companies for too long. Furthermore, the Data Protection Act is just a baseline for the protection of personal data: there may be reasons to implement additional privacy protection measures.

The public transport companies and TLS feel that there is no market for a privacy-enhanced version of the OV-chipkaart. When asked, users might choose a privacy-enhanced card over a regular one, but they are not willing to pay additionally for such a card. Also, changing the currently existing system would be a very costly operation, so there is no upside to changing the system. Moreover, TLS and the PTCs perceive the privacy debate as open-ended: besides the Data Protection Act, there is no universally agreed upon set of requirements for good privacy practices. This means that a concession to the privacy activists would not, in any way, improve their opinion of the system: they would only ask for more privacy protection measures.

U-Prove enables Verifiers to minimize the amount of data they collect about their customers, and Provers to interact with Verifiers without any possibility for the Issuer to find out. However, as the Issuers and the Verifiers are tied together closely and are possibly even the same party, the Issuers and Verifiers do not perceive a reason to distrust each other. The vision of TLS and the PTCs with regard to U-Prove mostly entails removing the need to trust the terminals at the gates and conductors, while maintaining the control that TLS and the PTCs currently have over traveller data.

One of the advantages of the OV-chipkaart that was presented at the introduction of the system, was that it might be possible in the future to publish anonimized passenger data. This data could then be used for academic research efforts into public transport traveller behaviour. The vision of Open Ticketing with respect to U-Prove strongly supports this goal, as it (by construction) only collects traveller data which contains no identifying information. However, the public transport companies have a strong commercial incentive to not publish this data: as most of the exploitation permits for trajectories of the Dutch public transport system are distributed by a public tender, a party has a significant

strategic advantage when it is the only one of the competitors who holds such detailed information about the travel movements of all its customers. Therefore, it is unlikely that such a possibility would, in itself, drive the adoption of a technology like U-Prove.

5.5 Discussion

Trans Link Systems is the party which is primarily responsible for the decision to make changes to the OV-chipkaart system. Therefore, we need to study their potential motivations for adopting U-Prove in order to gauge its potential to succeed.

Legal The primary motivation for TLS and the PTCs to adopt any privacy enhancing technology is a legal one. As they use the Data Protection Act as a benchmark for their privacy measures, it can be expected that a more urgent formulation of the requirement to minimize the amount of data which is collected, can actually interest these parties in a technique like U-Prove. However, the current Data Protection Act was purposely not explicit about the measures which should be taken, in order to give judges room to evaluate the implementations in the light of currently existing techniques and technology[11]. Also, suddenly changing this legislation to force such changes in the system does seem ill-advised. Such a change can be driven from the political field, but should be orchestrated in close consultation with TLS. A first step in such a process might be to limit the ways in which TLS may use the rich data in research and analysis of travel behaviour.

Commercial As mentioned before, there are two commercial reasons which keep TLS from implementing more extensive privacy enhancing technologies. The first is a perceived unwillingness of customers to spend more on a privacy-enhanced product, which would result in very limited revenues. While this is probably true, a pilot project for such a privacy-enhanced version would go a long way in improving the public standing of TLS with respect to privacy. Furthermore, Open Ticketing has already pointed out that it is by no means necessary to replace all OV-chipkaarts with privacy-enhanced versions: only the people who are actively interested in protecting their privacy might actually buy one. The second reason is the costs of implementing such a system, for which especially the research costs would be very high. However, some of these costs are already borne by the Open Ticketing foundation and the academic world. If TLS would commit itself to producing a privacy-friendly card, this would be a world-wide first. In the light of the current attention for privacy matters, additional funding might be, for example, a grant from the European Union.

[11] A document [28] exists which is more explicit in its recommendations, but it is dated (2001).

6. CASE STUDY: ENIK

> Enfin, il est nécessaire, vu les circonstances qui en commandent l'application, que le système soit d'un usage facile, ne demandant ni tension d'esprit, ni la connaissance d'une longue série de règles à observer.
>
> *La Cryptographie Militaire*, Auguste Kerckhoffs

In this chapter, we will perform a case study of the eNIK, the proposed electronic Dutch identity card. We will discuss the different uses of such a card and the view that the stakeholders have of the matter. We will also touch upon several legal issues with different approaches to this system. To conclude, we summarize the fundamental conflicts which arise when suggesting a particular implementation for the eNIK.

6.1 Environment

In the offline world, the use of an identity document issued by the government to prove some part of one's identity is ubiquitous. For example, a storekeeper asks a customer to show his identification when he wishes to purchase cigarettes, in order to verify his age. Most such documents contain many security measures through which anyone can verify its validity, e.g. by optically inspecting it. Therefore, if somebody presents an identity document for which the photograph matches his facial features, this forms a reliable verification of the identity of the person presenting the document. Online, many interactions require the assumption of an identity, for example registering at a website, or choosing a nickname for an online game. In the last years, the online and offline worlds have become increasingly intertwined, leading to an increased use of aspects of one's offline identity (your real name, photograph, date of birth, etc.) in online interactions. In some cases, this behaviour is not just a custom, but a legal or commercial requirement. Financial transactions and e-commerce are two fields in which such a link is imperative in order to enable detection and prosecution of fraud.

Currently, many online companies possess extensive profiles about their users. For social networks, this is even intrinsic to the functionality that they offer to their users. Technologies such as OpenID allow such a company to act as an identity provider for its users. Concretely, this means that a user can reuse the identity he has at this website at other websites as well. For example, if John wishes to comment on my blog, I may ask him to login using his Facebook

identity. He does not need to register at my website, which is a convenience for John. Furthermore, my knowledge of John is now much more intricate, as I have a link between his comment and his identity on Facebook. Obviously, this poses two risks. First, Facebook did not verify Johns identity in any way when he registered. It is quite possible that the person who claims to be John actually is somebody else entirely. Second, the linking of these two identities may present a considerable privacy concern for John. Not only do I know much more about him than when he had chosen an arbitrary pseudonym or 'nickname', but Facebook also registers this interaction, thereby enriching the profile they already have about John.

In an attempt to resolve both these issues (trustworthiness of online identities and privacy implications of using a single identity provider), multiple commercial parties have proposed identity management solutions. For example, Digidentity[1] provides an online locker service, which is used to store identity information. The company can associate any number of identity attributes with your locker, such as phone numbers, e-mail addresses, the information from an identity card or a PKI certificate. Now, if you wish to disclose some part of your identity to a third party, you give Digidentity permission to provide this third party with a specific part of your identity which is stored in the locker. The third party needs to trust Digidentity to have accurately established the identity elements which they share about you. Also, you need to trust Digidentity to not secretly register the third parties to whom you have provided (parts of) your identity information. Such a system might provide solutions to most previously mentioned concerns. However, many stakeholders, such as owners of online shops, want to have direct confirmation of the identity of a customer from the government [27]. This would require the translation of the notion of a government-issued identity card to the digital domain.

The most straightforward analogy of the offline situation in the online world, would be some form of digital or virtual identity document, which is issued by the government. The authority of the government with respect to your identity is already trusted as a matter of course by virtually all third parties. The cryptographic mechanisms to transfer this testament of authenticity onto a set of identity attributes are already in place: a digital signature would show that the government has indeed issued that particular set of attributes. Two important questions remain in this scenario. First, not every third party needs (or indeed, should have) access to all attributes which are signed by the government. How do we make sure that this party does not access attributes for which it has no authorization? Second, if a government employee is able, through legal or illicit technical means, to retrieve the information users have shared with an online service, he can infer which citizens have used this service. This potential online omniscience of the government is generally viewed as undesireable or, at least, as an impairment of citizen privacy. Is there a way in which we can avoid such a scenario?

[1] `http://www.digidentity.eu`

6.2 Existing structures

The Dutch government provides a centralized online authentication system, DigiD[2]. This system is designed to be used as the authentication service of citizens to any government service which requires identification of its users. It provides three levels of authentication, Low, Medium and High, which correspond with increasingly strict authentication mechanisms, such as a username/password combination or a PKI certificate. Each government service defines under which circumstances a certain level of authentication is required. After the appropriate authentication mechanisms have been applied, DigiD provides the government service to which the user is authenticating with the Burger Service Nummer (BSN)[3] of the user. Subsequently, the government service may contact government databases to read, store or modify additional information about the identity of the user. An example of such a database is the Gemeentelijke Basisadministratie (GBA)[4].

Using this functionality in third party applications seems straightforward. After all, DigiD can authenticate the user and provide the third party service with the BSN of the user. The third party would then contact a government database such as the GBA and retrieve the relevant identity attributes. However, there is a legal reason why this approach is not possible: processing the BSN of citizens is illegal for anybody who has no explicit right to do so. This right is only granted to government agencies and health service providers and insurances. The reason for this limitation is that the BSN forms a unique and persistent identifier for a citizen. Linking existing profiles across databases would be straightforward: third parties could connect the profiles they have generated by comparing the BSNs in both databases. Furthermore, access to the GBA is strictly controlled. Allowing anybody to retrieve information from this administration would have serious implications for citizen privacy. Effectively securing non-government access to this information would be at best very costly and probably all but impossible.

Recently, the Ministry of the Interior and Kingdom Relations of the Netherlands has begun a project which is aimed at providing each citizen with an electronic identity card. This solution is named 'eNIK' for 'Elektronische Nederlandse Identiteitskaart'. Currently, the government is researching the feasibility of introducing such a card [14]. In the second half of 2011, the political process surrounding this decision will commence, which might eventually lead to the introduction and broad adoption of such a card. In this chapter, we study several possible approaches to implementing such a card, and we discuss the role that U-Prove might play.

[2] `http://www.digid.nl`

[3] The BSN is the successor to the sofinummer, the social security number. It is an identifying number which is unique for each citizen. The government is required by law to use this number when determining the identity of a citizen [4].

[4] The Gemeentelijke Basisadministratie is the municipal citizen administration of the Netherlands.

6.3 Involved parties

Ministry of the Interior The government branch which is responsible for the introduction of the eNIK is the Ministry of the Interior. This ministry is currently developing a proposition about the introduction of this card. In this proposition, multiple requirements will be put forward with regard to the functionality and security of the eNIK. One of the main concerns of the ministry is the requirement that the card should be easy to use by everybody: not only IT experts, but also the very old, very young and the mentally or physically disabled. This puts significant strain on the steps which may be required of the user of such a card: if using the card is too complicated, it will not be used, which will hamper the broad adoption of this type of online identities. In other words, if an online party requires it, every citizen should be capable of proving his identity using his card.

Furthermore, the ministry formulates three requirements which describe the functionality of the card. These requirements are the following:

1. The card should be usable in the non-government domain, with strictly enforced limits on the amount of information that a third party receives. This means that a citizen should be able to use the card at, for example, an online shop to prove some aspects of his identity. As the law currently prohibits the use of the social security number or BSN outside the government domain [4], this means the card should not allow third parties to process the BSN.

2. Using the card in the non-government domain (i.e. by third parties) should not allow the construction of identity hotspots by these third parties. An identity hotspot is a database in which the information in multiple profiles of a user is linked to create even richer profiles. The BSN would be a straightforward attribute along which such profiles could be matched.

3. It should not be necessary or possible to change the contents of the card after its issuance. This is proposed as an additional security measure, which seems to stem from an urge to avoid a situation such as the hack of the OV-chipcard (mentioned in the previous chapter).

These three requirements seem unreconcilable. Indeed, when using 'classical' (i.e. non-zero knowledge) cryptographic techniques, at least one of the three will not be achieved. U-Prove does not provide a solution to this problem either: In order to prevent creation of hotspots, one needs an unlimited number of tokens, which contradicts the third requirement. IBM has developed a cryptographic system, Idemix[5], which is similar to U-Prove, but which can satisfy all three requirements at once. Idemix uses zero-knowledge proofs to a much greater extent than U-Prove, which makes it more complex and computationally intensive. On the other hand, it does provide cryptographic guarantees which U-Prove does not.

[5] `http://www.zurich.ibm.com/security/idemix/`

Third parties In the offline situation, many parties use government-issued identity documents to identify and authenticate customers. In the online world, it is even harder to identify the party with whom communication occurs. Therefore, even more parties will require a document which enables identification and authentication of clients in an online environment. Thuiswinkel.org, the Dutch representative organisation of online shops, will be used as a representative of all third parties in this context. They have elaborated on their position in [27], from which we derive the following requirements of an online identity card like the eNIK:

1. The government should issue authenticated (e.g. digitally signed) statements about the identity of customers such that a third party may rely on that statement from a legal point of view.
2. The authenticated statements of the government should be auditable, i.e. it should be possible for an observer who was not involved in the authentication process to assure himself of the fact that assurance has actually taken place. Preferably, such an audit does not require the disclosure of sensitive data (keys, customer information) to any other party than the auditor. For example, it should not be necessary for the government to decrypt certain data before it becomes auditable.
3. One of the main and growing concerns of online shops is fraud: in 2011, an estimated 1.9% of online transactions in the Netherlands was conducted in a fraudulent manner (1.6% in 2010)[7]. A way in which this could be combated is accurately identifying each customer and prosecuting all instances of fraud. Therefore, the eNIK should be suitable to combat fraud by enabling the owner of an online shop to report fraudsters to the police.

Thuiswinkel.org has stated other requirements for such an identity service, but these are from a convencience perspective: we have only considered the requirements which directly benefit from the government functioning as the identity provider.

Civil rights groups The notion of identity is an important one in many debates concerning civil rights. Because the government has unparallelled investigative powers in the online domain, it is generally considered prudent to limit the amount of information that it gathers about its citizens to the minimum required to achieve its goals. In particular, it is required that the government can not find out to which third party a citizen has disclosed his government-authenticated personal information.

Also, privacy is often defined as "(...) the claim of individuals, groups, or institutions to determine for themselves when, how, and to what extent information about them is communicated to others." [29]. Therefore, empowering citizens by allowing them to decide themselves to what extent and with whom they wish to share their government-authenticated personal information is considered a fundamental requirement from a privacy perspective. Moreover, even if a citizen would agree (e.g. through ignorance or commercial pressure) to disclose more information than the third party is legally allowed to process, this disclosure should be prevented through technical means in the eNIK.

6.3.1 Proposed solutions

This subsection describes four different ways in which the eNIK could provide identity services to third parties. In the next section, we will study the advantages of each of these solutions in the light of the requirements which were formulated by the different stakeholders. Note that these proposals are our own, are academic in nature and do not necessarily reflect the view of the involved parties.

Chip with BSN The most simple and straightforward approach mimics the functionality of DigiD: store an encrypted and signed copy of the BSN of the owner of the card on the chip. If the owner needs to authenticate himself, the eNIK outputs this encrypted and signed value of the BSN. The owner forwards this value to the third party to whom he is authenticating himself. Now, the third party can verify the digital signature of the government to check that the value was actually issued by the government. He will now contact a government database such as the GBA to retrieve further information about this customer. There are limits to the type of data which he is able to retrieve from this database is limited: the limits are controlled by the government and may vary due to legislation or the type of party which contacts the database. For example, an authenticated claim that a customer is over 18 may be relevant to an online liquor store, but not to an online music store. Although the actual BSN of the customer is hidden in this way, it is still identical for each authentication at each store. Therefore, this approach does not prevent the construction of hotspots. Also, any store can reuse this encrypted BSN at any other time in the future. A possible way to solve this is through the use of salting and timestamping: at each authentication, a random value, the name of the store and a timestamp are added to the plaintext BSN before encrypting it on the eNIK. In this way, the store can not, from its communication with the eNIK, re-recognize a client. Also, the access control system of the database can now be certain of the timeliness of the request: It is only giving out data at the moment that the store was actually authorized by the data subject (the person identifying himself) to receive this information.

Signed attributes An obvious downside of the previously stated solution is that the government is aware of every piece of information which was looked up through the eNIK of the user. An alternative solution, which avoids this particular downside, is the following. All the attributes (full name, date of birth, photograph, BSN, etc.) which are to be stored on the eNIK are signed by the government and subsequently stored on the chip of the eNIK before it is issued to the owner. Now, if the owner of the eNIK wishes to show some part of his identity to a third party, he requests the signed set of attributes from the card, which he forwards to the third party. This party can verify the digital signature of the government on the attributes, so he knows that this set of values was indeed issued by the government. There is no need to contact the government database in this scenario: all relevant information is already stored on the eNIK. However, note that there is no way to limit what information is disclosed: as the government signs the entire set of attributes, no selective disclosure can occur, otherwise the third party can not verify the digital signature of the government. Using separately signed attributes does

not resolve all problems, as these signatures are still unique, which results in the possibility of a hotspot. Also, separating these attributes poses a risk, as it might enable illicit recombination of these attributes: combining signed attributes from multiple cards, a man in the middle might be able to construct new identities which are not only not his, but do not exist at all.

U-Prove, one token In many cases, the legislation about the collection of personal information is not absolute. As long as there is a stated goal for the collection and there is consent of the person about whom personal information is gathered (the data subject), the gathering party is usually within legal bounds. However, verifying that the data subject has actually consented to the collection is hard to implement with a centralized database infrastructure. Also, the 'signed attributes' solution does not offer any way in which too extinsive data collection can be prevented. U-Prove might present a solution to this problem.

In this solution, the government plays the role of the Issuer of U-Prove tokens in which the attributes of a citizen are encoded. Each eNIK contains one such token, and plays the role of the Prover. Now, if the owner of the eNIK wishes to disclose some aspect of his personal information, he allows the third party which requested this information to interact with his eNIK. The third party plays the role of the Verifier. The owner of the eNIK authorizes it to generate a presentation proof which states the aspects of his attributes which he wishes to disclose. After the presentation proof is generated and sent to the third party, the third party can verify the proof, i.e. it can check that the proof was generated using a U-Prove token which was issued by the government and which contains attributes for which the statements in the proof actually hold. Due to the zero-knowledge property of the proof (see Chapter 4 for details), it is impossible to infer anything about any undisclosed attributes which are encoded in the token. Note that in this scenario, the government is only involved at the issuance of the eNIK with the U-Prove token. Afterwards, no interaction with the government is necessary, so there is no way in which the government can easily uncover who disclosed what information to which third party. Moreover, even if the government was to retrieve the presentation proofs which the third parties have stored, they are not able to link these to a specific issuance moment. Therefore, the government can, as the Issuer, not uncover anything more from a U-Prove token than the third party can uncover by itself.

U-Prove, multiple tokens A downside of the previous solution is that it uses only one token. As mentioned previously, this means that all the presentation proofs which are generated using that one token can be linked to each other: with each presentation proof, the public key of the token is sent along, which is unique for each token. If you prove at one online store that you are over 18, and in another one that your first name is John, these two stores can collude to find out that the customer whose name is John is the same as the customer which is over 18. In other words, identity hotspots can be constructed. U-Prove does not provide a way to circumvent this: proofs that have been generated by using the samen token will always be linkable to each other through the public key of the token. In that sense, a solution which uses multiple tokens is preferable.

If we modify the previous solution by storing multiple tokens on the eNIK, some other things change as well. First, which token should be used to generate a presentation proof for each request? A straightforward way to implement this is by using a different token for each third party. The list of which token is used for which party is kept on the eNIK as well. This would potentially require a great number of tokens. A more economic approach might be defining different 'identity domains' which are able to create smaller hotspots. For example, I could use my eNIK to prove that I am over 18 to multiple websites which contain adult material. Now, I could choose to use one token for these websites and nothing else. The consequence would be that all these websites not only know that I am over 18, but that they could combine their data to determine that they all had the same visitor. The advantage in this case is that I would not be embarrassed about this disclosure. Rather, I would be embarrassed if this information could be linked across identity domains, so that my employer or my doctor finds out. This would reduce the number of tokens which are required, but there still is no upper limit to the number which I might need. Either I should be able to dynamically add or replace tokens on the card by an interaction with the Issuer (i.e. 'charging' the eNIK at a government-controlled terminal), or I should be able to choose which token I use by myself when the interaction occurs. These two approaches can also be used in tandem. Note that letting the owner of the eNIK choose which token to use might be much too complex for many citizens. Also, many citizens will not understand why they are 'charging' their identity card. In the table which summarizes these solutions, we assume a solution in which the eNIK can be 'charged' and the owner can decide how he wants to use these tokens, either on the fly or by pre-defining identity domains.

In both U-Prove solutions, the eNIK should be able to determine what legal bounds apply to each information request. U-Prove itself does not describe ways in which a Prover might authenticate the information request of a Verifier. A straightforward approach to this problem is to let the government (e.g. a newly established independent supervisory board) authorize the information requests of third parties. For example, if an online store wants to request the date of birth of customers, they would formulate a query which eNIKs can interpret. This query could then be approved and digitally signed by this board. An eNIK will now be able to verify that this party is indeed legally allowed to request this information from a customer. In this way, no illegal information requests could occur.

Requirement	Chip with BSN	Signed attributes	U-Prove, one token	U-Prove, multiple tokens
Ministry of the Interior				
No identity hotspots should be constructable by third parties	Constructable by linking encrypted BSNs	Constructable by linking all info	Constructable by linking U-Prove token	Not constructable
Third parties should be able to use the authenticated information in a *limited* way	GBA can perform rudimentary access control	All info is always shared	Amount of info sharing can be accurately controlled	Amount of info sharing can be accurately controlled
The contents of the smartcard should be fixed	Contents are fixed	Contents are fixed	Contents are fixed	Card can be 'charged'
The system should be easily usable by every citizen	No citizen interaction needed	No citizen interaction needed	Citizen chooses attributes to share	Citizen chooses attribute and token to share
Third parties				
The law should allow third parties to use the authenticated information	BSN processing is prohibited [6]	BSN processing is prohibited	Legal limits can be accurately enforced	Legal limits can be accurately enfored
The proofs received from a user should be auditable	The encrypted BSN could be stored outside the card	The attributes could be stored outside the card	Only the card can generate (auditable) proofs	Only the card can generate (auditable) proofs
The system should be suitable for fraud prevention	Using the BSN, law enforcement can trace fraudsters	All info is available	No revocable privacy	No revocable privacy
Civil rights groups				
The government should not be able to find out to whom I disclose my attributes	Government is notified of every use	Law enforcement might collude with third parties	The government has the same information as the third parties	The government has the same information as the third parties
It should be impossible for a third party to retrieve more attributes than they are allowed to legally	Can be enforced at the database (but how to do so exactly is undefined)	No limit on the amount of information gathered is imposed	Only authenticated requests	Only authenticated requests
It should be impossible for a third party to retrieve more attributes than they are allowed to by the user	No user interaction whatsoever	No user interaction whatsoever	User can limit information sharing to an arbitrary measure	User can limit information sharing to an arbitrary measure

Tab. 6.1: Summary of the extent to which the proposed solutions fulfill the requirements of the involved parties (green: Fully compliant, yellow: Partially compliant, red: Not compliant).

6.4 Conflicts

Ministry of the Interior requirements As mentioned previously, the Ministry of the Interior has formulated three requirements for the eNIK which are hard to combine: usable by third parties, no hotspot construction possible, and a card with only fixed contents (i.e. non-chargeable). The Idemix system which was designed by IBM seems to be the only viable way to achieve this. However, experimental implementations of Idemix on a smartcard show it to be much slower than U-Prove, as a result of the additional computational complexity. Also, using Idemix would require all attributes which are stored on the card to be fixed: possible future use cases where third parties are allowed to store their own attributes on a card are, in such an implementation, impossible. Therefore, we will consider what the consequences are of ignoring any one of these requirements. If the usability by third parties is dropped, the eNIK will be restricted to the government domain, which is a severe restriction of the usability of the card. If the construction of hotspots is allowed[7], the one-token U-Prove solution is the obvious choice for all stakeholders. However, this solution has serious implications for the privacy of citizens, as their movements can be tracked across different identity domains. More specifically, if a user uses his token to share identifying information[8] with a government service just once, the government can track this user in all third party databases to which they have access through the investigative powers of police and intelligence agencies. Allowing the eNIK to be chargeable with additional government or third party tokens could resolve these issues. It is impossible to create a smartcard which can be guaranteed to be unbreakable now and in the future. However, smartcards are already used for electronic wallet systems, which seems to imply that commercial parties can adequately secure important data.

Multiple tokens: essential freedom or total confusion? For anybody who is proficient with computer systems, the multiple-token solution for eNIK is ideal: it gives the user full freedom to choose how he wants to arrange his online identity. For people who are not as experienced with such systems, the question "Which token do you wish to use for this identity proof?" is daunting and unexpected: they would expect such a system to 'just work' and not bother them with decisions which they were not required to make in the offline equivalent of the eNIK. It is an open question how a user can be properly involved in the decision on which token to use for each interaction, while still maintaining usability for less experienced users of the system.

Illegal use of BSN The BSN is a unique and persistent identifier for every citizen. These three properties (unique, persistent, everybody has one) make it a highly sensitive identity attribute. This is why its use is strictly controlled by law [4]. In short, processing the BSN is allowed only in the government domain, with very few exceptions (most notably, the health care domain). Using the BSN or an equivalent number therefore poses significant limitations on the domain of allowed use of the eNIK, which results in an unwelcome restriction of the

[7] Note that preventing hotspot creation is stated as a ministry requirement, but that it is obviously shared by the civil rights groups

[8] For example, the BSN is identifying. Also, in many cases the full name of a citizen is (almost) unique, and therefore identitifying

usability of the card. Therefore, the first and second solution are not viable in the current legal context.

6.5 Discussion

Ubiquity of smartcard readers Even with the increasing popularity of PKI, for which the use of smartcards is an important facilitator, the number of citizens with access to a smartcard reader is still relatively small. However, to be able to use the eNIK in online transactions, one needs a smartcard reader. Moreover, if a third party wishes to *require* the use of eNIK for his service, smartcard readers should be ubiquitous. All solutions for the eNIK system which we have proposed, depend on access to a smartcard reader by every citizen. Possible strategies to achieve the situation in which everybody has access to a smartcard reader should be considered, as the success of this technology heavily hinges on it. Also, the choice for regular or contactless smartcard will influence this availability, as most readers can handle only one of the two types. Interestingly, the Belgian government has already introduced a (regular) smartcard-based identity card in 2003, but by 2010, only 29% of citizens had access to a smartcard reader at home[9]. An alternative approach may be to search for other smartcard-like platforms, such as NFC, of which it is expected that they will reach the required market penetration. In an NFC scenario, one could use his telephone to identify himself using the eNIK functionality, with the added benefit of the availability of a semi-trusted user interface (namely, the operating system of the phone) to enable selection of the token and the attributes to disclose.

Allowed questions by a Verifier As noted above, there are legal limitations to the amount of information which third parties may request and process from an eNIK. Especially in the online world, these limitations should be enforced, as the user awareness of online privacy risks is low, and some third parties may be hard to trace once illegal gathering of identity information is discovered. According to us, the best approach to this is to let an independent supervisory board or trusted third party regulate the questions that a third party is allowed to ask. Already, it is compulsory to register any personal information processing with the CBP, although this is often 'forgotten' by the parties which are responsible for this processing [3]. Such a structure could ensure that the processing of personal information will actually be more closely watched and that it will be harder to collect too much information accidentaly or on purpose.

Perception by the public When comparing the offline and online uses of an identity card, we see a difference. In the offline case, showing your identity card implies that your identity is completely disclosed to the party which inspects your card. In other words, your privacy in the sense of [29] (privacy as control) ends when using the identity card. In the online case however, U-Prove or similar technologies might allow citizens to actively and minutely control the amount of information they share with different parties. Therefore, the identity card promotes limiting the amount of information a service provider has of you. We speculate that difficulties may arise in the public's perception of the possibilities of a U-Prove-enabled eNIK. They might extrapolate their offline experiences to

[9] `http://www.itprofessional.be/trendsentips/112882/eid-onderschat-maakt-onbemind/`

an online expectation: Since I am using my identity card, I can no longer expect any amount of privacy in this interaction. Such a stance means that the public will not appreciate the added privacy features of the eNIK. Moreover, they will then view any limitations on the amount of personal data a service provider might request as unnecessary and cumbersome.

7. CASE STUDY: DEFENSIEPAS

In this chapter, we will discuss the possibilities for using U-Prove in the Defensiepas system, which is the proposed identity card system of the Ministry of Defense. One of the main difficulties of this environment is the number of other parties at which the card should also be usable. Furthermore, the development of Defensiepas is still in a relatively early stage, which means that not all stakeholders and their positions are obvious yet. We discuss a possible approach which, however, is more hypothetical than in the previous chapters, as a result of the still somewhat malleable environment.

7.1 Environment

The Dutch Ministry of Defense is currently in the process of replacing their smartcard access system with a new system. This system, which is called 'Defensiepas', will provide authorization assertions within the domain of the Ministry of Defense. However, the personnel of the Ministry of Defense has to cooperate frequently with other organisations, such as the UN, NATO, Interpol or non-military sectors of the Dutch government. This card will also serve authorization purposes in a cross-domain context, where the Ministry of Defense grants one of their employees an authorization which can be interpreted by a relevant other (supra-)governmental body. Understandably, not all these bodies can be persuaded to change their system: the system which is introduced by the Ministry of Defense will have to be compatible with the currently existing systems.

7.2 Requirements of Ministry of Defense

When comparing this case study with the previous two, we find a significant difference in the privacy requirements of the systems. In the case of the OV-chipkaart and eNIK, the primary party which demanded privacy-friendly alternatives for currently existing or proposed solutions was the end user: the traveller or citizen. In both cases, personal considerations led to a demand for stronger privacy protection techniques. In the case of Defensiepas, the main reason for requesting stronger privacy protection is safety. Many of the personnel members of the Ministry of Defense play a strategic role in the national security of the Netherlands. Therefore, it is important that it is hard to track the location or the movements of high-ranked ministry personnel. A compromised smartcard reader or a malicious eavesdropper should not be able to infer with whom he is interacting from the electronic communication which he observes. For example, the Dutch military has several security clearances, which have letters. From lowest to highest, these are E, D, C, B, A and A+. This clearance

system is hierarchic, so everybody with a high clearance is automatically authorized to perform any action which somebody with a lower clearance is also authorized to perform. From the perspective of the verifier of an authorization, it is unimportant what the exact security clearance is of the person with whom he interacts: the important information is whether the security clearance is *at least* of the required level. A low-security gate should in no way be able to recognize a high-ranking general from a common soldier. As we have seen in previous chapters, U-Prove provides a way in which to achieve this, using interval proofs.

The content in this chapter is organized differently from the content in the previous two chapters. The Defensiepas system is still in an early stage of development, which means that it is hard to predict the consequences of certain design choices such as using U-Prove. Furthermore, the environment in which the Defensiepas will be applied is highly complex: many different parties rely on the validity of a proof, which means that interoperability with existing systems is an important requirement. However, since the environment is so diverse, we were not able to make an accurate prediction of the problems which might arise when using U-Prove in such a case. Therefore, as we were not able to make a full assessment of the situation in this case, we describe the answers which we did find and attempt to distill insights for the possibility of adoption of a technology such as U-Prove for a relatively new and complex system such as Defensiepas.

Methodology We acquired most of the information about Defensiepas in an interview with Lieutenant Colonel Fekke Bakker, who is responsible for innovation projects with respect to identity management in the Ministry of Defense. As we also noted in the previous chapters, while we present the ideas we have discussed in the interview, we might not have interpreted his words correctly.

7.3 Observations

When discussing the possibility of using U-Prove for Defensiepas, one of the recurring questions was the measure in which U-Prove was a 'open specification'. As noted earlier, Microsoft has released U-Prove under its Open Specification Promise, which does allow for non-Microsoft implementations, although it does not rule out the possibility of infringing patents of third parties. Therefore, it is not exactly clear whether U-Prove is 'open enough' for the purposes of the Ministry of Defense. The two main reasons for asking for an open specification are as follows:

- It allows the Ministry of Defense to verify the cryptographic specifics without having to trust the owner of the specification or to resort to non-disclosure agreements.
- Should Microsoft decide not to continue with the U-Prove system, the Ministry of Defense can hire any other programmer to build a system which implements U-Prove. Also, Microsoft may decide to continue with the U-Prove system, but exclude the Ministry from using the system. Such

dependence can be a risk for national security, but it may be mitigated by legal or procedural controls.

Clarifying the exact freedoms which an implementing party has with respect to U-Prove might prove instrumental in the adoption of U-Prove in domains which require the certainty of prolonged support from a supplier.

Another interest of the ministry is the cryptographic guarantees which U-Prove offers, and their strength when compared with a policy-based approach. We briefly describe such a policy-based approach. The guarantees which U-Prove achieves cryptographically can also be achieved by using a trusted third party. If the Prover trusts the systems which are run by the Issuer and Verifier, he does not have to be as reluctant to disclose his information. Obviously, the cryptographic guarantees which U-Prove offers are stronger than the policy-based promises of the Verifier or Issuer, for two reasons. First, they might have a considerable (business) interest in covertly collecting your data, thereby breaking their own policy. Second, trusting an Issuer or a Verifier is not enough to trust their systems. A malicious intruder might have compromised their systems, thereby being able to access Prover information to which he is not privy in the policy-based case. The cryptographic approach (i.e. U-Prove) does not have this weakness. Therefore, in a high-security domain, the strength of the cryptographic guarantees of U-Prove make it a better choice over a policy-based system.

7.4 Opportunities

In the current state of development of Defensiepas, it is hard to be specific about the way in which U-Prove might be applied to improve the privacy protection of the system. We therefore present a hypothetical approach to applying U-Prove here.

The environment of Defensiepas is a typical case of federated identity management: multiple organizations all have their own means of identifying their employees, and while the authorizations of a particular individual may span more than one organization, only one is authorative with respect to the security clearances of this individual and making assertions about these to other organizations. An important standard in this field is SAML, Security Assertion Markup Language, which provides a common language in which to formulate assertions about the authorizations of individuals.

For example, an employee of the Ministry of Defense who has security clearance B has to visit a NATO office in The Hague. The NATO and the Ministry of Defense have an agreement about visitors to the NATO building: anyone with a security clearance of C or higher is allowed to visit without a previous temporary approval. Now, if this individual indeed has the clearance, he presents proof of his identity to the access control of the NATO building (for example, a turnstile at the entrance which is equipped with a smartcard reader). The NATO systems now request an assertion from the Ministry of Defense about the identity of the employee: does he indeed have a security clearance of at least

C? The Ministry of Defense checks their own employee registration system and finds that this is the case: the employee has security clearance B. He responds with an assertion which states that this employee has a security clearance which is appropriately high for these purposes. Satisfied with this answer, the NATO systems authorize the entrance of the individual, and the turnstile opens.

Now, what could be potential privacy issues here which U-Prove might be able to resolve? The main problem is leaking the identity of the individual: there are no measures which ensure that his identity can not be stored separately by the NATO systems or a malicious third party which is able to communicate with the smartcard of the employee. Once this identity is stored, any assertions made about this identity provide more information for an attacker. Now, as soon as somebody has been identified as an individual with a high security clearance, he might be subjected to spear phishing, identity theft, bribery, coercion or blackmail, in order to access the information which he holds. Typically, such individuals are also potential strategic targets for a terrorist attack.

A related yet smaller problem is the lack of possibility for offline authentication. When an individual presents his card at the entrance, the access control system needs to contact the party which manages the individual's identity and security clearance in order to assert whether the individual should be allowed access. One can imagine several cases in a military context in which this is a particular problem, such as on a submerged submarine or at a very remote base in the mountains or jungle. In such cases, it would be useful to be able to rely on the information which the smartcard provides when making access control decisions.

Implementation As before, we make a suggestion for a U-Prove implementation by identifying the Issuers, Verifiers and Provers in this context. We resolve the aforementioned problems by implementing U-Prove on the Defensiepas, the smartcard which is distributed among all the personnel of the Ministry of Defense. We have constructed this implementation in such a way that it can be built within the currently existing SAML context.

The role of the Prover is played by the Defensiepas which contains several U-Prove tokens. These tokens have been issued by the Issuer, which is the department of the Ministry of Defense which is also responsible for physically issuing the Defensiepas itself. These tokens are all constructed using the same set of attributes. Therefore, they are in principle all capable of constructing proofs for the same statements. However, the smartcard software ensures that each token is only used for interval proofs of the type 'The security clearance of this employee is at least #', where # is fixed per token. Furthermore, the tokens for the higher security clearances are used for a limited number of proofs. In a sense, each token represents a separate pseudo-identity, which is more security-sensitive when the clearance for which it is used is higher. Depending on the available amount of storage on the Defensiepas, it will occasionally be necessary to issue new tokens for an existing Defensiepas. The role of the Verifier will be played by the access control system of the building which the employee wishes

to access: this may be a facility of the Ministry of Defense, but it might also be one of the federated identity management partners of the ministry.

The procedure which should be followed for revoking a Defensiepas is not entirely straightforward. After all, we can not store a unique identifier on each Defensiepas, otherwise the effect of using multiple psedo-identities will be negated. Microsoft has announced that it intends to design a revocation mechanism in a next version of U-Prove, which might be based on the current idea of a hidden attribute for which the Prover proves that it is *not* included on a given revocation list. Note that, as we described in Chapter 3, such a proof is not currently possible.

Design and development When compared to the previous case studies, the implementation which will be used for the Defensiepas system is in an early stage of development. In a sense, this presents both a problem and a possibility for U-Prove as a candidate technology in this case.

Most large IT projects begin with defining the requirements of the product or system which will be delivered at the end. These are not just functional requirements: the privacy requirements are also set at this stage. The problem for U-Prove is that it does not only provide traditional privacy guarantees: rather, it allows for privacy protection which is based on a drastically revised idea of privacy, which centers around the notions of pseudo-identities and defining privacy as control. Since many who are responsible for defining the requirements for a new IT project are not familiar with this way of thinking, the privacy requirements which are stated are largely aligned with the possibilities which are offered by traditional PET's. If these requirements can be fulfilled by an older technology, the possibilities of U-Prove will be viewed as superfluous and overly resource-consuming.

On the other hand, in order to fully appreciate the possibilities that U-Prove offers, one needs to redefine traditional notions of identity and privacy. Only at the beginning of a project can we expect to find support for such a paradigm shift: in order for a technology like U-Prove to become a success, the project requirements should require much higher standards than are required today by law, uninformed consumers and well-intentioned but traditional and time-pressured project managers.

8. DRIVING FACTORS

In the past chapters, we have studied many aspects of applying U-Prove in different scenarios. Throughout these studies, we have identified many points which are relevant to the decision whether to choose U-Prove for a particular implementation. In this chapter, we summarize these points and, in some cases, formulate them in a more general manner. Then, we discuss three different high-level approaches to implementing U-Prove in a system. These ideas stem from our experiences while performing this research.

8.1 Summary of aspects

We have divided the aspects which we have found between four different domains: legal, societal, commercial and technical. Obviously, the distinction between these domains is not always clear.

Legal factors

- The Dutch Data Protection Act [3] does require 'adequate' protection measures for the storage and processing of personal data. However, it is not explicit in its requirements, which means that it does not require a technology like U-Prove.
- The Dutch law on use of the BSN [4] doesn't allow the BSN to be used in non-government domains. This means that it is illegal for a commercial party to use this unique identifier to retrieve further information from a government database. Any information it needs will have to be retrieved using other means, for example, by selective disclosure through a U-Prove-enabled smartcard.
- In many cases, the law requires that certain interactions are auditable: an external party should be able to ascertain himself that a specified interaction has indeed occurred. Often, this is hard to achieve in the presence of far-reaching anonymization techniques. However, using non-interactive proofs, U-Prove does allow for such a requirement. Sometimes the law requires more information about the identity of the parties involved in an interaction than would be strictly necessary in a cryptographical sense: in such scenarios, the law would have to change before U-prove can provide the actual guarantees which make it worthwhile.

Societal factors

- Selective disclosure of attributes can be achieved either through software and organizational processes or cryptographic means. It is unclear which

approach is preferred in general: in high-security environments (e.g. the ministry of defense), the cryptographic approach is preferrable. However, in cases where user perception of privacy or authentication is important, it may be hard to convince the user that U-Prove actually provides the guarantees which the user requires. In such a case, a combined or software/organization-only approach may be more appropriate.

- Civil rights and privacy advocacy groups often demand improvements with regard to privacy protection of personal data in commercial or government systems. However, the owners of these systems need an external benchmark to determine when their privacy protection is good enough. They perceive the privacy debate as open-ended: no matter what measures they implement, the privacy advocates will always want more.

- Implementing U-Prove on a smartcard which is issued by the party who is also the Issuer of the U-Prove tokens on it has some drawbacks. For example, the user of such a smartcard can not be sure of the way the software on his smartcard works, which hardly would mean an improvement in the perceived privacy of users. Furthermore, even if U-Prove is implemented conscientiously, it is important that the user is accurately aware of which attributes are disclosed in which interaction.

- Since the Issuer is often responsible for the architecture of the U-Prove system as well, the inability to re-recognize the tokens which he issues may seem like an unwanted feature of U-Prove. After all, the Issuer trusts himself, even if the Provers do not. Furthermore, he will probably be curious to the interactions of Provers with different Verifiers.

- People associate using an identity card with losing all anonimity. Explaining to them that using U-Prove actually keeps their anonimity intact may prove impossible.

- In order for U-Prove to become successful in some scenarios, it is important that smartcard readers become ubiquitous. It is unclear how this can be achieved, but it is an important premisse for the eNIK case, and useful at least in the OV-chipkaart and Defensiepas case.

- When people think about the privacy protection which they want, they formulate their requirements with respect to the possiblities which they know exist. Since U-Prove provides new possiblities, people are unlikely to come up with such requirements on their own, thereby slowing down the adoption of both this broader notion of privacy protection and the adoption of privacy protecting technologies such as U-Prove.

Commercial factors

- Commercial parties have indicated that they require stronger authentication of their clients in order to combat fraud and be able to offer a broader range of services. They see the government as a candidate which might offer this kind of authentication of citizens.

- Many forms of data analysis on databases of personal information depend on the richness of this information. This goal is diametrically opposed to the goal of data minimalization for which U-Prove strives. It may be possible to effectively perform market research on the pseudonymized data, but it is not at all clear how to do so.
- In the view of some companies, privacy protection is not a selling point with which one can draw a large market share. This means that they feel no direct commercial incentive to adopt U-Prove.
- It is expected that in the near future a law will come into effect in the Netherlands which requires reporting all data leaks to the involved parties, including the users of which the identities were leaked. If such a law comes to exist, there will be a much greater incentive towards data minimalization, as the reputation loss will be considerable in such a case.
- Data minimalization and anonymization allow for the disclosure of large databases of operational information for (academic) research purposes. However, in the case of heavy competition, e.g. in a public tender, such operational information is closely linked to a commercial advantage, thereby negating the incentive to serve the public interest by disclosing this information.

Technical factors

- In its current form, the U-Prove specification does not state any ways in which to achieve token or user revocation. Some suggestions are made, but all case studies show that this is an important feature.
- More complex proofs (e.g. 'my attribute is *not* equal to' or 'the sum of these attributes is equal to x') are not possible (yet).
- Generating interval proofs and complex presentation proofs (many undisclosed attributes) are slow operations on a smartcard, but not nearly as slow as Idemix proofs.
- Each U-Prove token allows for exactly one pseudo-identity. If one needs a new pseudo-identity, one needs to get a new token from an Issuer. As a corrollary, this means that when all Issuers collude, they know an upper limit to the number of pseudo-identities any person has. Depending on the implementation, by refusing to issue new tokens to replace old ones, the Issuers may be able to control the number of pseudo-identities per user.
- In more complex environments, multiple Issuers are each responsible for a small part of the information which is encoded in attributes about a user. This means that the information across multiple U-Prove tokens should be linkable, for example by each incorporating some link identifier for which they prove that they are equal (without disclosing the link identifier). However, it is currently unspecified and unclear how to achieve his.

- Many parties, especially in the government domain, rely heavily on the openness of the system: they need to be able to scrutinize the cryptographic specifics of the system and its implementation. They also need to be sure that their implementation will work with the implementations of other parties, and that they do not depend on one company for all their maintenance and support with respect to this technology.

8.2 Three high-level approaches to implementing U-Prove

While researching the case studies in the previous chapters, we have identified a few approaches to introducing U-Prove into a new or existing project. Each has several advantages and disadvantages: there is no set 'best approach' for introducing U-Prove into a system. However, choosing one of these approaches allows for a consistent view on the possible advantages that U-Prove has to offer for a project.

U-Prove as an add-on In the case study of the OV-chipkaart, we studied an existing system which might be improved when combining it with U-Prove. As a result of discussing this possibility with the privacy officer of NS, we described a possible approach which amounted to changing some of the cryptographic specifics of the current implementation of the OV-chipkaart by U-Prove tokens and presentation proofs. When you do not change the architecture of the system or the underlying identity concept, this is the best you can achieve: you can exchange your current cryptographic mechanisms for U-Prove.

Since the notions of Issuer, Prover and Verifier in the sense of U-Prove are artificially introduced in this scenario, the trust relations which are enforced in U-Prove are considered superfluous and are, when necessary, already enforced elsewhere through software or organizational procedures. Since the legal provisions in the Netherlands are not technology-specific, no legal shortcomings of your implementation will be resolved by this replacement. When compared to your original implementation, the version with U-Prove may be a considerable improvement with regard to end-user privacy, but this depends both on the specifics of the original implementation and the ways in which you add U-Prove to this implementation. The advantage of choosing this approach is that it is relatively cheap and easy to apply, that it requires almost no changes to systems which are not directly in contact with the U-Prove token and that it leaves most of your organizational processes intact.

U-Prove in the architecture In the Open Ticketing view of the U-Prove version of the OV-chipkaart, extensive changes were made to the underlying architecture: responsibility for different parts of the information stored on the card suddenly became divided between several parties. The fraud detection system was also restructured to allow for anonimity-preserving fraud detection. In short, the goals and requirements of the OV-chipkaart were taken and the system was restructured in such a way that the goals and requirements were all met with the strongest privacy guarantees for the end-user.

In this approach, U-Prove plays a central role in designing the system architecture. Since every part of the system may be redesigned with maximum end-user privacy in mind, this will be a favorite for privacy advocates and civil rights groups. On the other hand, redesigning a system leads to radical changes to most of the elements in the architecture: in the OV-chipkaart example, the IT systems of several parties needed to be upgraded or replaced, the fraud detection process was replaced and made possibly less effective and the overall performance (with regard to check-in times) was probably hampered. Furthermore, in many cases such a redesign will have implications for the level of service that an organization can offer to end-users: if it no longer has access to the information which it had before (due to data minimalization), it may not be able to effectively help users who experience problems. A traveller who wishes to view his latest travels can now log on to the OV-chipkaart website and view this information. If the redesign is actually implemented, he will require a smartcard reader to do so. Moreover, such a redesign requires a significant commitment from the management level, as it will have extensive financial and functional repercussions. Properly explaining the advantages of using U-Prove to these people may be difficult, especially since most organization do not (yet) perceive any strong commercial drivers for the adoption of such a privacy enhancing technology.

U-Prove as an identity paradigm The two previous approaches take the currently existing system and U-Prove, give one of the two the lead (existing system in the first case, U-Prove in the second) and try to mix the two. But what if we go even further? In the eNIK case study, we have seen that using a classic approach to designing such an identity card suffers from severe limitations. We have proposed two U-Prove based solutions, but these are not adaptations of a previously existing structure. Rather, they redefine the way such an identity card attests the identity of its owner, by introducing multiple pseudo-identities which can all be separately and selectively disclosed.

Such a solution requires a different approach, a different way of thinking about identities. Not only should the system allow a single user to use multiple pseudo-identities, it should actively try to discourage the notion of actual users and only work with the presented pseudo-identities. This is the most in-depth approach to empowering users with regard to their identities, and it has several drawbacks. For example, the notion of a pseudo-identity is not yet natural to many users, which means that they do not perceive the necessity nor the advantages of using pseudo-identities in their online activities. This would make a U-Prove based system both superfluous and difficult to use. Furthermore, this change requires the entire organization to think differently about identities and the possibilities of linking a pseudo-identity to a natural person: for commercial (research), legal (auditability) and technical (functionality) reasons, management and other personnel will not welcome such a change in the system. However, if this concept of pseudo-identities is widely adopted, it will allow for easier interoperability between different Issuers and Verifiers, such as in the federated identity management scenario which was sketched in the Defensiepas case study.

Both the Defensiepas and the eNIK case study have the potential for such a change of perspective, as they both have significantly different decision dynamics than the OV-chipkaart case study. In the case of Defensiepas, the requirement for end-user privacy is not primarily felt by the end-users themselves. Rather, the Ministry of Defense, the issuer of Defensiepas, perceives privacy of its personnel as instrumental for national security. This in itself can be a strong driver towards U-Prove as an identity paradigm. In the case of eNIK, the Ministry of the Interior may not be primarily interested in all the privacy guarantees of U-Prove, but the ultimate decision with regard to the introduction of eNIK and its privacy requirements lies with the Dutch Parliament, which represents exactly the end users for whom the privacy guarantees of U-Prove are indeed relevant.

9. CONCLUSION

> "Het is gezien", mompelde hij, "het is niet onopgemerkt gebleven". Hij strekte zich uit en viel in een diepe slaap.
>
> *De Avonden*, Gerard Reve

In this thesis, we have answered the research question which was posed in the introduction. We have studied U-Prove from multiple perspectives in order to assess its potential for success, and we have presented multiple factors which are of influence to this success. Here, we formulate some conclusions to this research, as well as some recommendations for further work.

In all three case studies, there was room for improvement of the current or proposed solution by using U-Prove. The approach we have taken here, namely identifying all the requirements and demands of the different involved parties, has proven to be an effective and transparent method towards assessing the viability of using U-Prove in a given system. Many of the limitations we see for the adoption of U-Prove are organizational: knowledge of the politics which surround the decision for introducing such a system will therefore be crucial in order to achieve such a transition.

The three case studies we have performed were based on using a smartcard to perform the cryptographic tasks of the Prover in the U-Prove specification. This allows for strong security guarantees, as the assumption when using a smartcard is that there are only limited ways in which one can access the information stored upon it. On the other hand, it also requires many computational resources, which are almost always limited on a smartcard. In context of the case studies we did, we think it is meaningless to use a smartcard to store the attributes when you perform the Prover calculations outside the card. A viable path would be to create a smartcard whose cryptographic coprocessor can perform U-Prove calculations directly, which would have a great positive impact on the performance of the system.

Even though the law indirectly suggests technologies such as U-Prove as solutions for identity and personal information management problems, it does not enforce it. In other words, all legal requirements can be fulfilled by using technologies which are less protective of user privacy. In the future, it is not expected that these provisions will change: only in the government domain is there a place for the user (i.e. citizen) to voice his concern, through his representation in the parliament. In the case of privately owned systems, one will

need to show his discontent by switching to another service provider or threatening to do so. Obviously, this is not always an option, as can be seen in the OV-chipkaart case study.

Companies do not, currently, view privacy protection as a way to attract more customers, decrease cost or increase revenue. Moreover, privacy protection is seen as detrimental to their ability to perform effective market research and deliver satisfactory customer service. Only when these problems are adressed alongside any technical difficulties will the introduction of U-Prove be viable in any organization for which privacy protection is not a primary concern. The compulsory reporting of data leaks, which is part of currently planned legislation, may become another incentive to this end, though it is still unclear whether customers will respond to data leaks by deciding to switch to another service provider.

Societally, there are two competing notions. On the one hand, privacy protection is increasingly perceived as an important goal which can only be achieved by exercising strict control on your personal information. On the other hand, any additional steps needed to authenticate oneself in the online domain are considered cumbersome and too difficult for less proficient users. It is unclear how such a paradox should be reconciled. It would seem that more research is needed to find ways in which user privacy can be protected without recquiring many difficult steps from the user himself.

All three case studies show that the current cryptographic specification of U-Prove is lacking in its support for a basic feature: token revocation. As long as this feature is not implemented in an efficient manner, not one of the interviewees will consider using U-Prove in his system. Additionally, more research could be done in order to realize the more complex proofs which are possible in the original mathematical description of U-Prove [5], but which are not yet in use in the current specification.

We believe that U-Prove is a viable product which has many good opportunities for improving end-user privacy in the online domain. On the other hand, we think that the discrepancy between which party benefits from the added privacy protection and which party decides about implementing U-Prove may lead to a neglect of technologies such as U-Prove, since no decision maker feels the need to choose such technologies over other existing, legally sufficient and commercially more viable, solutions. This discrepancy can be resolved in one of two ways. First, a large party such as a government can feel political pressure towards protecting user privacy in an effective and definitive manner, and take the lead in building a system which spurs the use of U-Prove among other parties (e.g. the eNIK case study). Second, Microsoft can use its significant market penetration to push U-Prove forward, not just as an open specification which everybody may use, study and extend, but also as an authentication system which is a viable alternative for currently existing technology and which moreover presents a unique possibility to adequately protect the privacy of one's users.

BIBLIOGRAPHY

[1] Wetboek van strafrecht, artikel 232, lid 1. http://lexius.nl/wetboek-van-strafrecht/artikel232/lid1. [Online; accessed 26-July-2011].

[2] Telecommunicatiewet. http://wetten.overheid.nl/BWBR0009950/geldigheidsdatum_26-07-2011, October 1998. [Online; accessed 26-July-2011].

[3] Wet bescherming persoonsgegevens. http://wetten.overheid.nl/BWBR0011468/geldigheidsdatum_26-07-2011, July 2000. [Online; accessed 26-July-2011].

[4] Wet algemene bepalingen burgerservicenummer. http://wetten.overheid.nl/BWBR0022428/geldigheidsdatum_26-07-2011, July 2007. [Online; accessed 26-July-2011].

[5] S. A. Brands. *Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy*. MIT Press, Cambridge, MA, USA, 2000.

[6] D. Chaum. Blind signature system. In *Advances in Cryptology - CRYPTO '83*, page 153, New York, 1984. Plenum Press.

[7] J. de Bel, J. Boersma, and A. Screpnic. Online betalen 2011. Report, Innopay, Thuiswinkel.org, February 2011. Version 1.0.

[8] A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew Odlyzko, editor, *Advances in Cryptology CRYPTO 86*, volume 263 of *Lecture Notes in Computer Science*, pages 186–194. Springer Berlin / Heidelberg, 1987.

[9] J.-H. Hoepman, B. Jacobs, and P. Vullers. Privacy and security issues in e-ticketing – optimisation of smart card-based attribute-proving. In Vitaly Shmatikov Veronique Cortier, Mark Ryan, editor, *Workshop on Foundations of Security and Privacy*, Edinburgh, July 2010.

[10] Microsoft. Microsoft U-Prove cryptographic specification. https://connect.microsoft.com/site642/Downloads/DownloadDetails.aspx?DownloadID=26953. [Online; accessed 4-February-2011].

[11] Microsoft. Microsoft U-Prove CTP. https://connect.microsoft.com/content/content.aspx?contentid=12505&siteid=642. [Online; accessed 4-February-2011].

[12] Microsoft. Microsoft U-Prove technology overview. https://connect.microsoft.com/site642/Downloads/DownloadDetails.aspx?DownloadID=26953. [Online; accessed 4-February-2011].

[13] M. Naor, Y. Naor, and O. Reingold. Applied kid cryptography or how to convince your children you are not cheating. `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.46.9932`, 1999. [Online; accessed 28-March-2011].

[14] Ministry of the Interior. Digitale veiligheid en identiteit — rijksoverheid.nl. `http://www.rijksoverheid.nl/onderwerpen/digitale-overheid/digitale-veiligheid-en-identiteit`. [Online; accessed 22-June-2011].

[15] T. Okamoto. Provably secure and practical identification schemes and corresponding signature schemes. In Ernest Brickell, editor, *Advances in Cryptology CRYPTO 92*, volume 740 of *Lecture Notes in Computer Science*, pages 31–53. Springer Berlin / Heidelberg, 1993. 10.1007/3-540-48071-4_3.

[16] College Bescherming Persoonsgegevens. OV-bedrijven bewaren gegevens reisgedrag in strijd met de wet. `http://www.cbpweb.nl/Pages/pb_20101209_ov-chip.aspx`, 2010. [Online; accessed 9-May-2011].

[17] C. P. Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, 4:161–174, 1991. 10.1007/BF00196725.

[18] B. Schoenmakers. Sharp interval proofs and other predicates. Unpublished manuscript, February 1997.

[19] B. Schoenmakers. Cryptographic protocols. Lecture notes of Cryptography 2, Technical University of Eindhoven, 2010. `http://www.win.tue.nl/~berry/2WC13/LectureNotes.pdf`.

[20] B. Schoenmakers. Personal communication, 2011.

[21] Trans Link Systems. Factsheet Geschiedenis van de OV-chipkaart. `http://www.translink.nl/backstage/media/bijlagen/algemeen/Factsheet_geschiedenis_invoering_OV-chipkaart_v1.pdf`. [Online; accessed 9-May-2011].

[22] Trans Link Systems. Privacybeleid Trans Link Systems. `http://www.ov-chipkaart.nl/pdf/22246/privacyovchipkaart`. [Online; accessed 26-July-2011].

[23] Trans Link Systems. Systeemarchitectuur - level 0. `http://www.translink.nl/backstage/media/bijlagen/algemeen/Factsheet_systeemarchitectuur_-_level_0_v1.pdf`. [Online; accessed 26-July-2011].

[24] Trans Link Systems. Systeemarchitectuur - overzicht. `http://www.translink.nl/backstage/media/bijlagen/algemeen/Factsheet_systeemarchitectuur_-_overzicht_v1.pdf`. [Online; accessed 26-July-2011].

[25] The European Parliament and the Council of the European Union. Directive 1999/93/EC of the European Parliament and of the Council of 13 December 1999 on a Community framework for electronic signatures. *Official Journal of the European Communities*, L 12:12–20, 2000.

[26] The European Parliament and the Council of the European Union. Regulation (EC) No 1371/2007 of the European Parliament and of the Council of 23 October 2007 on rail passengers rights and obligations. *Official Journal of the European Communities*, L 315:14–41, 2007.

[27] Thuiswinkel.org. Identiteitsdiensten; vijf aanbevelingen. Position paper, Nederlandse Thuiswinkel Organisatie, Ede, June 2011. Version 1.0.

[28] G. W. van Blarkom and J.J. Borking. Achtergronden en verkenningen 23. `http://www.cbpweb.nl/downloads_av/av23.pdf`, April 2001. [Online; accessed 26-July-2011].

[29] A. F. Westin. *Privacy and freedom*. Atheneum, 1970.

APPENDIX

A. EXTENDING U-PROVE WITH INTERVAL PROOFS

> A woman can keep one secret - the secret of her age.
>
> Voltaire

In Chapter 4, we have introduced a new type of presentation proof for U-Prove tokens. The regular presentation proof allows for selective disclosure of some of the attributes which are encoded in the token. Proofs of the new type, the *interval proofs*, convince a Verifier that the Prover possesses a token for which a specific attribute is contained in a certain given interval. In this appendix, we give background information for these interval proofs, which mostly consists of mathematical specifics regarding their derivation.

The public key of a U-Prove token is of the following form:

$$h := (g_0 g_1^{x_1} \dots g_n^{x_n} g_t^{x_t})^{\alpha} . \tag{A.1}$$

The regular presentation proof, which is used to disclose a subset of the attributes encoded in this public key, amounts to using a zero-knowledge protocol which convinces the Verifier that the public key is indeed of this form and the disclosed attributes are indeed encoded into this public key. This protocol is used non-interactively: by using the Fiat-Shamir heuristic, the proof can be constructed without the regular three-way communication between Prover and Verifier.

The interval proof we present consists of three steps:

1. Rewrite and split the public key of the U-Prove token so that the subproofs can be constructed.
2. Apply the basic interval proof twice, as described in Section A.2.
3. Apply the non-interactive Okamoto identification protocol to the remaining part of the public key, as described in Section A.3.

Each of these steps correspons to a step in the verification process, which consists of verifying the separate proofs which the Prover has generated.

A.1 Rewriting the U-Prove token public key

In order to apply the basic interval proof and Okamoto proof algorithms to the public key of a U-Prove token, we need to rewrite the initial expression. In

[18], a splitting technique is used to split a form with multiple exponents to a form with only two exponents. This technique can not be applied in this case however, as the public key has an additional exponent α. Therefore, we use a rewriting technique to achieve this (courtesy of [20]).

Let j be the index of the attribute for which we wish to show that it is contained in an interval $[a, b)$. Assume that $e_j = 0$, so we can easily move from A_j to x_j and back. Now, to prove that x_j is contained in the interval $[a, b)$, we rewrite the public key expression A.1 to the following equivalent form:

$$h^{\frac{1}{\alpha}} = g_0 g_1^{x_1} \dots g_n^{x_n} g_t^{x_t}$$

$$g_0 = h^{\frac{1}{\alpha}} g_1^{-x_1} \dots g_n^{-x_n} g_t^{-x_t}.$$

Let $\alpha' = \frac{1}{\alpha}$ and $\forall i : g_i' = \frac{1}{g_i}$. Then the expression becomes

$$g_0 = h^{\alpha'} g_1'^{x_1} \dots g_n'^{x_n} g_t'^{x_t}.$$

We use the splitting technique described in [18] on this expression to create two expressions. Let $\alpha_1', \alpha_2' \in_R \mathbb{Z}_q$, with the constraint that $\alpha_1' + \alpha_2' = \alpha'$. Now, we split the expression into $g_0 = g_0^{(1)} g_0^{(2)}$ with

$$g_0^{(1)} = h^{\alpha_1'} g_1'^{x_1} \dots g_{j-1}'^{x_{n-1}} g_{j+1}'^{x_{n+1}} g_n'^{x_n}$$

$$g_0^{(2)} = h^{\alpha_2'} g_j'^{x_j}.$$

We use these two expressions for our further calculations. Using the Okamoto proof technique, we generate a proof which shows zero-knowledge that we possess all the exponents in $g_0^{(1)}$, namely the x_i-values except x_j. Let

$$k = \min\{\gamma \in \mathbb{Z}_{\geq 0} \mid b - a \leq 2^\gamma < q\}.$$

Now, we note that showing that

$$x_j \in [a, b)$$

is equivalent to showing that

$$x_j \in [a, 2^k + a) \text{ and } x_j \in [b - 2^k, b)$$

since, by definition of k, it holds that $2^k + a \geq b$. Using the basic interval proof technique, we generate two proofs. One shows that

$$x_j - a \in [0, 2^k),$$

which is equivalent to

$$x_j \in [a, 2^k + a).$$

The other shows that

$$x_j - b + 2^k \in [0, 2^k),$$

which is equivalent to

$$x_j \in [b - 2^k, b).$$

Therefore, the combination of these proofs convinces a Verifier that $x_j \in [a, b)$.

A.2 Basic interval proofs

The basic interval proof we present here is based upon the interval proof which was presented in [18]. The interval proof presented there applies to expressions of the form

$$B = g^{\alpha} h^{x}, \tag{A.2}$$

for which it convinces the Verifier that the value of x lies in a predetermined interval $[0, 2^k)$. We will call such interval proofs *basic interval proofs*, as they form the basis upon which our 'full' interval proofs are built. Since [18] is an unpublished manuscript, we present the main idea here as well.

In [18], the following protocol is presented which achieves the aforementioned goal[1]. First, the Prover constructs k subcommitments $B_i = g^{\alpha_i} h^{x_i}$ with $x_i \in \{0, 1\}$ and $\alpha_i \in \mathbb{Z}_q$ such that $x = \sum_{i=0}^{k-1} x_i 2^i$ and $\alpha = \sum_{i=0}^{k-1} \alpha_i 2^i$. Then, for each subcommitment, the Prover executes the protocol depicted in Figure A.2. After performing the verification steps for each subcommitment B_i, the Verifier checks whether reuniting the subcommitments actually yields the original commitment B:

$$B = \prod_{i=0}^{k-1} B_i^{2^i}.$$

As this protocol is interactive, it is not suitable for generating non-interactive proofs which can be verified by any Verifier. Therefore, the author suggests using the Fiat-Shamir heuristic to construct a challenge for this protocol. We do this by replacing the random challenges c_i which are chosen by the Verifier by the following:

$$c \leftarrow \mathcal{H}(B, a_{0,0}, \ldots, a_{k-1,1}).$$

Note that this challenge is calculated after all the commitment steps for all the subcommitments have been executed, and that it is the same for all subcommitments. As calculating a hash function can be quite computationally intensive, reducing the number of hashing operations can have a significant impact on performance. After this replacement, the protocol can be performed by just one party, the Prover, so it can be used to generate a non-interactive proof which is sent subsequently to the Verifier, or it can be used as part of a larger proof construction. If it is part of a larger proof construction, only one challenge is necessary for all of the separate proofs: the challenge can then be calculated as the hash of even more publicly available values.

A.3 Non-interactive Okamoto identification protocol

The Okamoto identification protocol [15] is a well-known zero-knowledge protocol. It applies to expressions of the form

$$h = g_1^{x_1} g_2^{x_2}.$$

The Prover runs this protocol with the Verifier to prove that he possesses the values x_1 and x_2. We modify the protocol to include more than two exponents

[1] There is an error in the protocol description in [18], which leads to an incorrect version of the protocol. In the version presented here, this error has been corrected.

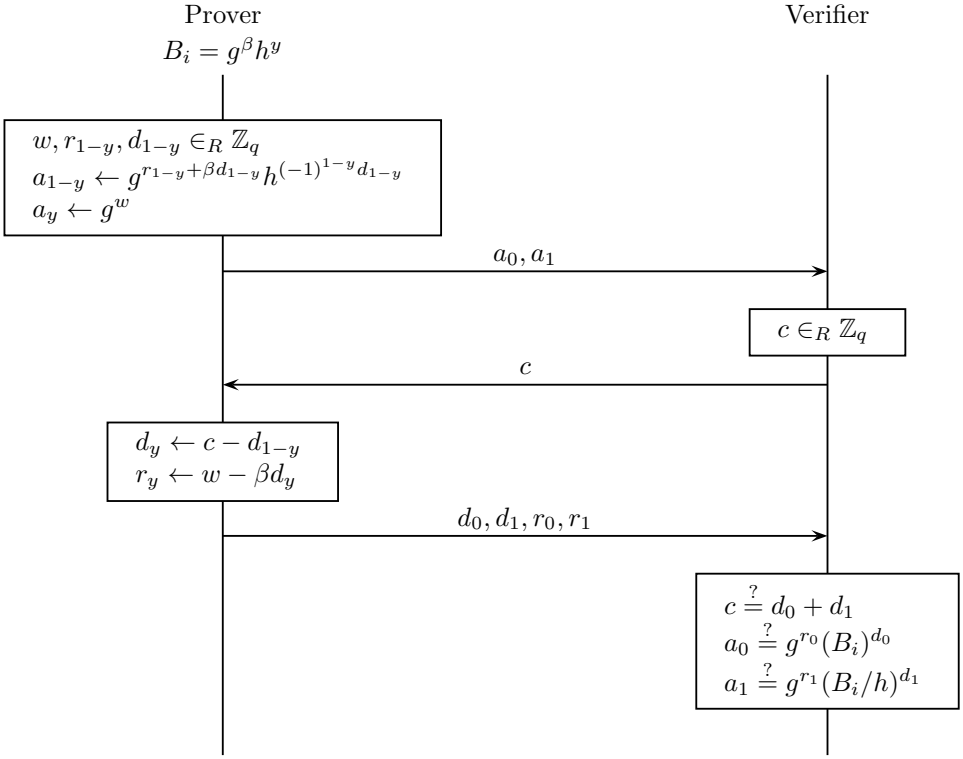

Fig. A.1: Proof that commitment B_i has the correct form. For readability, we have set $\beta = \alpha_i$ and $y = x_i$.

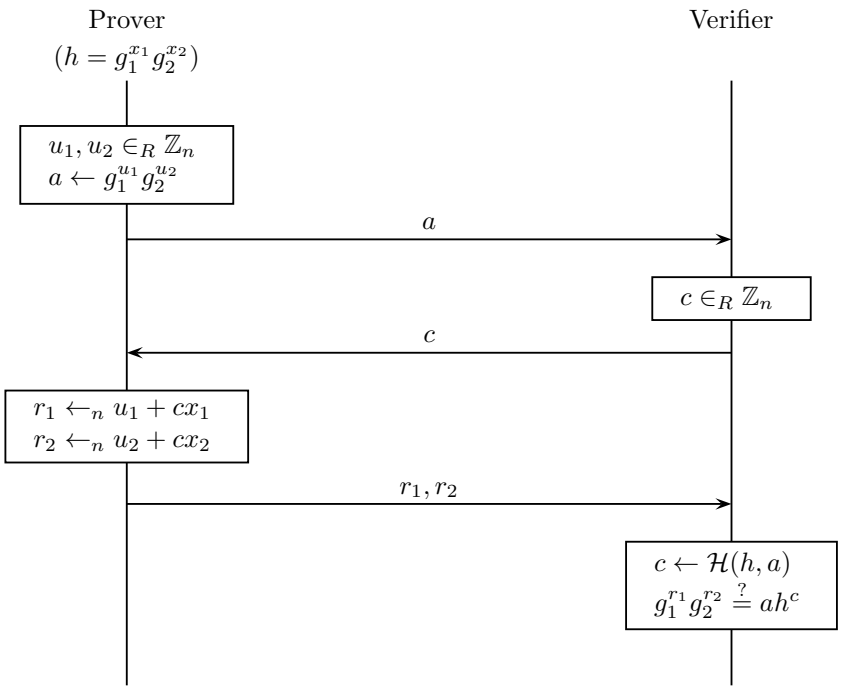

Fig. A.2: Okamoto's identification protocol

and to make it non-interactive. The original protocol is depicted in Figure A.3:

Expanding the protocol to more than two variables is straightforward: we calculate additional u_i and r_i as required, and return all the r_i upon completion. In order to apply the Fiat-Shamir heuristic in this protocol, we replace the challenge generation step by

$$c \leftarrow \mathcal{H}(h, a).$$

After this replacement, the protocol can be performed by just one party, the Prover, so it can be used to generate a non-interactive proof which is sent subsequently to the Verifier, or it can be used as part of a larger proof construction.

A.4 Implementation of interval proofs in the Microsoft SDK

In order to demonstrate certain capabilities of the U-Prove technology, Microsoft has released an experimental SDK for U-Prove. Following their example, we have implemented the additional capabilities which we designed into the context of this SDK. The extended SDK is based upon version 1.0 of the U-Prove SDK, and it can be downloaded at `http://rogaar.org/thesis/`

`UProve10WithIntervalProofs.tar.gz`. In Table A.4 we summarize the changes which we have made to the SDK in order to implement the new features.

File name	**New?**	**Additions/modifications**
BasicIntervalProof.cs	Y	Contains all functions necessary to generate and verify basic interval proofs
GroupDescription.cs	N	Added code to calculate an inverse in the group G_q (using the extended Euclidian algorithm)
HashFunction.cs	N	Added code to allow hashing of a two-dimensional array
IntervalProof.cs	Y	Contains all functions necessary to generate and verify interval proofs, including the proper calls to the subalgorithms
OkamotoProof.cs	Y	Contains all functions necessary to generate and verify Okamoto proofs
SDKSample.cs	N	Added code which demonstrates the proper way to call the interval proof functions

Tab. A.1: Description of file changes to the Microsoft U-Prove SDK