User–Centric Identity Using *e*Passports

Martijn Oostdijk, Dirk-Jan van Dijk, and Maarten Wegdam

Novay, P.O. Box 589, 7500AN Enschede, The Netherlands
{martijn.oostdijk,dirk-jan.vandijk,maarten.wegdam}@novay.nl

Abstract. The worldwide introduction of *e*Passports presents a unique opportunity for the online identity community to implement trustworthy identity providers. The *e*Passport provides citizens with a strong authentication token within a global Public Key Infrastructure backed by government administrations. This paper studies the possibilities for leveraging the *e*Passport for user-centric identity and reports on an experiment in which *e*Passports are combined with the user-centric identity management framework Information Card. Note that no changes to already deployed *e*Passports are needed for our solution to work.

1 Introduction

Most online services (*e*Commerce, *e*Government) are only meaningful if (aspects of) the identity of users can be established in a trustworthy manner by whoever offers the service. At the same time users will only use a service when they feel that they are in control over who they share identity information with. This paper investigates, through a practical experiment, how strong authentication means (i.e. *e*Passports) can be combined with user-centric identity management.

Over the last couple of years electronically readable travel documents (*e*Passports) have been introduced in most countries of the world. An *e*Passport contains an embedded chip with card holder data which allows an automated inspection system (typically operated by border control officials) to read out data from the chip and, more interestingly, to verify the integrity of the data and the authenticity of the chip. The embedded chip communicates in a contactless manner based on standardized communication protocols, ensuring that the chip can be contacted when it is in the proximity of an inspection system.

While there are concerns about the privacy consequences of the introduction of *e*Passports [5,8,10,16,17], primarily caused by the combination of contactless communication with privacy sensitive biometric data, it also presents a unique opportunity for creating trustworthy online identities as it potentially provides citizens with a strong authentication token within a global Public Key Infrastructure (PKI) backed by government administrations [9]. Moreover, the technical standards which describe how the inspection system verifies the authenticity of *e*Passports are open and publicly available from the International Civil Aviation Organization (ICAO[1]) [6]. Although originally not

[1] See http://icao.int/

Y. Chen et al. (Eds.): SecureComm 2009, LNICST 19, pp. 296–310, 2009.
Institute for Computer Science, Social-Informatics and Telecommunications Engineering 2009

intended as such by ICAO, *e*Passports, as they are being deployed presently, seem ideal for authenticating users of third-party online services such as web stores.

At the same time an entirely different revolution is taking place in the online identity community which places the end-user at the center by relaying all communication between identity providers and service providers (also called relying parties) through the user's client. Web 2.0 services are driving this revolution, which is therefore sometimes dubbed Identity 2.0 or user-centric identity, and it is being enabled by identity management systems such as OpenID [13] and Information Card [12].

The objective of the research described in this paper is to study the possibilities for leveraging the *e*Passport for user-centric identity. This would establish online identities backed by government issued hardware tokens in a user-centric manner. This paper describes a prototype user-centric online identity solution based on Information Cards which uses *e*Passports for authentication. The implementation makes it possible for users to show aspects of their *e*Passport (aspects such as information stored in the chip's memory but also proof of authenticity of the chip) to relying parties (online service providers) with the help of an identity provider. The identity provider only needs to store minimal information about the user's passport and can be seen as a privacy filter from the user's perspective.

The solution is built using open standards and is published as open source software.

The remainder of this paper is organized as follows: Section 2 presents a brief introduction to the ICAO standards and describes the various features of the *e*Passport. Section 3 introduces user-centric identity management and in particular the Information Card framework. Section 4 explains how these standards are combined into a working prototype information card identity provider that uses *e*Passports for authentication and lists some of the results. Section 5 discusses the lessons learned. Section 6 presents concluding remarks and pointers for future work.

2 The ICAO *e*Passport

This section provides a short introduction to the ICAO standard for *e*Passports, for more details see [5,9,16].

The ICAO standard for *e*Passports is described in Doc 9309 [6], which itself is based on many other standards and specifications. The specification consists of two parts:The first part describes the format of the contents of the *e*Passport, the so-called Logical Data Structure. The second part describes a variety of mandatory and optional security controls which are implemented by the *e*Passport to protect the information in the Logical Data Structure against various forms of attack.

2.1 Logical Data Structure

The contents of an *e*Passport are structured in terms of so-called data groups. Together with an index file (COM) and a signature file (SOd) these form the Logical Data Structure. Table 1 lists the data groups found in a typical Dutch *e*Passport.

Table 1. Contents of the Logical Data Structure

COM	An index of which DGs are present
DG1	The contents of the MRZ (name, date of birth, …)
DG2	JPEG or JPEG2000 image of face
DG11	Optional passport holder's full name if too long for MRZ
DG15	Public key for Active Authentication
SOd	Security document with signature over Logical Data Structure hashes

Two files are always present: COM contains an index which indicates which of the 16 possible data groups are present. SOd is the security document which contains the issuing country's signature over the contents of the data groups. It contains hashes for each of the data groups present in the Logical Data Structure and a signature over these hashes. This allows Passive Authentication as described in Section 2.2.

The first data group, DG1, contains the textual information about the passport holder that is also (optically) printed in the Machine Readable Zone (MRZ) on the data page of the *e*Passport. The information in DG1 contains the passport holder's name, date of birth, gender, as well as the document's number and date of expiry. In the Dutch case DG1 also contains the passport holder's citizen number (the Dutch equivalent of a social security number).

The public key in DG15 is used for a security mechanism called Active Authentication which is described in Section 2.2.

For the purposes of this paper the Logical Data Structure elements of interest are: DG1 which contains textual information about the passport holder and the document itself, DG15 which contains a public key, and the SOd which contains a signature over the different data groups.

2.2 Security Controls

To protect against attacks such as skimming, altering, unauthorized access and cloning the *e*Passport contains a number of security controls.

Basic Access Control: When attempting to read the Logical Data Structure, the *e*Passport requires the inspection system to first show knowledge of an access key comprised of three items in the MRZ: the passport document number, the date of birth of the passport holder, and the date of expiry of the passport. By requiring the inspection system to prove knowledge of these items, the passport is convinced of the fact that the inspection system has seen the data page of the physical passport booklet, which means that whoever is operating the inspection system has access to the booklet and has the passport holder's consent to read it. BAC prevents skimming in which an attacker gets access to an *e*Passport without the holder's knowledge or consent.

Extended Access Control: Some data groups contain information of a highly sensitive privacy nature, such as biometric templates. To protect against unauthorized

parties reading such files an additional access control mechanism may be implemented on top of BAC. Whether an inspection system can complete the EAC protocol when presented with an *e*Passport depends on whether it has acquired a document verifier certificate (DVC) from the *e*Passport's issuing state.

Passive Authentication (PA): The security document (SOd) attached to the Logical Data Structure contains hashes of all data groups and a signature over these hashes. The signature is set using a Document Signing Private Key and can be checked using the Document Signing Public Key Certificate (DSC), which in European Union passports is included inside the SOd. The DSC, in turn, is signed using the Country Signing Private Key and can be checked using the Country Signing Public Key Certificate (CSC). This latter certificate, at least in the Dutch case, can be downloaded from a government website. PA prevents altering the data in the Logical Data Structure (either by changing or replacing the chip or by intervening with the communications between chip and inspection system).

Active Authentication (AA): The inspection system can challenge the chip to prove authenticity by signing on request a random nonce using a document specific private key. The corresponding public key can be read from DG15 (which is part of the Logical Data Structure, and therefore part of the signed data in the SOd) so that the inspection system can check the resulting signature. AA prevents cloning, as the private key cannot be extracted from the *e*Passport by an attacker. The verification algorithm for AA is specified as an ISO standard [7].

For European Union passports BAC is mandatory. EAC is not widely used presently, but will be as soon as fingerprints are included in *e*Passports across Europe (expected mid 2009). PA is mandatory for all ICAO *e*Passports, however not all European Union member states allow third party access to their CSC. AA is optional and only a few countries implement it.

2.3 Software for Accessing *e*Passports

The ICAO standards have been implemented by various countries and manufacturers of identity products since 2005/2006. Open source initiatives to read *e*Passports soon followed, mostly with the purpose to test the various official implementations. Shortly before the introduction of the Dutch *e*Passport in 2006, software was developed at Radboud University to test the Dutch implementation of the *e*Passport [5]. Some of the results were later disseminated as an open source project, JMRTD (http://jmrtd.org), which we used in our prototype[2]. The software consists of a framework for reading and verifying passports using off-the-shelf hardware as well as a reference implementation in Java Card of the *e*Passport itself.

The JMRTD API offers data structures for the information stored in the Logical Data Structure, making it possible to interpret the data. The API also implements the various security controls used by the passport such as BAC, PA, and AA.

[2] Other open source implementations are the OpenMRTD project (http://openmrtd.org) and the RFIDIOt project (http://rfidiot.org)

3 User-Centric Identity

This section introduces user-centric identity management and focuses in particular on the Information Card specification [12].

Online identity management is a game for three: the user, the relying party, and the identity provider. The user wants to use a service provided by the relying party. At the same time the relying party wants to have some assurance about the client's identity. The identity provider helps the user and the relying party in providing this assurance. Sections 3.1 and 3.2 explain in more detail how this game is played.

Whether the user (with the help of the identity provider) succeeds in convincing the relying party that the claimed identity is correct depends on the level of trust that the relying party has in the identity provider. At the same time, the user also needs to trust the identity provider to only use information rendered for the purpose of acquiring the service from relying party. This privacy problem is discussed in Section 5.2.

User-centric identity management approaches place the user (contrary to e.g. the identity provider) in the center of the solution, which includes among others that the flow of information goes via the user. We use here the Laws of Identity created by Kim Cameron [4][3] to further define user-centric identity management. For brevity we only list them:

1. User control and consent
2. Minimal disclosure for a constrained user
3. Justifiable parties
4. Directed identity
5. Pluralism of operators and technologies
6. Human integration
7. Consistent experience across contexts

There are two prominent specifications for user-centric identity management: OpenId and Information Card. OpenID is a lightweight approach to user-centric identity that has its origin in 2005 for preventing spam through blog post comments. It is specified by the OpenID foundation[4]. Several open source initiatives exist[5]. For this paper however we used Information Card since it enforces privacy sensitivity through user-centricity, and is in the process of becoming a more formal standard.

The original Information Card specification is by Microsoft, and is called the Identity Selector Interoperability Profile. It was drafted with the above laws in mind, and the Information Card adheres to them. This specification was used as input to the Organization for the Advancement of Structured Information Standards (OASIS), which is now in the process of standardizing Information Card. The main factors of the Information Card specification that contribute to adherence to the Laws of Identity are the use of a card metaphor, and the routing of all identity claims through the user's client, as we explain below.

---

[3] An interesting side-note about Cameron's paper: One of the laws (the law of directed identity) is illustrated with a self-service passport reader example.

[4] See http://openid.net

[5] See http://wiki.openid.net/Libraries

The user interacts with the system through a so-called *Card Selector* (or Identity Selector). The selector presents the user with a number of visual cards containing claims (called fields) about the user's identity. The selector is a metaphor for a wallet containing all sorts of plastic cards.

All traffic between identity provider and relying party is routed through the user's client, literally putting the user in the center. This keeps the user informed and moreover gives the user the option of aborting the transaction at different stages of the authentication process.

Microsoft's CardSpace is a closed source card selector embedded into the Windows operating system. Microsoft's .NET framework offers building blocks for constructing identity providers and relying parties. Open source alternatives for card selectors, identity providers, and relying parties exist[6] as well. Some of the Information Card implementation details in this paper are CardSpace specific, and may be handled differently by other implementations.

3.1 Enrolling at the Identity Provider

Enrollment is the process of registering a user's identity with the identity provider. In Information Card enrollment results in a so-called managed card which (in the user's experience) is retrieved and made accessible in the Card Selector. Information Card allows two different types of cards: Self-issued cards contain claims made by the user about the user. Managed cards, on the other hand, contain claims by an online identity provider about the user. Retrieving a managed card is done by selecting some authentication mechanism which is used for proving possession of the managed card in the future. The managed card is said to be backed by that authentication mechanism. Typically a self-issued card is used as backing for a managed card but other options, like traditional username and password or an X.509 certificate corresponding to some private key (possibly on a hardware token), are also possible.

3.2 Using a Managed Card to Authenticate at the Relying Party

After the user has retrieved a managed card from the identity provider he or she can start visiting relying parties. Upon such a visit the relying party sends a policy back to the user's client which, amongst others, contains field names (such as "Last name" or "Date of Birth") deemed necessary by the relying party before the service can be acquired and specifies which identity provider the relying party trusts. The user is now typically presented with the Card Selector which shows only those cards which comply with the relying party's policy. The user selects one of those cards and in case it is the managed card the identity provider is called upon to fill in the values of those fields specified in the relying party's policy. The result is a so-called *security token* (generated by Security Token Service, a component of the identity provider) which is first sent back to the client.

Obviously, the token contains sensitive information and its confidentiality needs to be protected. To do this, it may be encrypted with the relying party's public key.

[6] See, for instance, the Higgins project at http://www.eclipse.org/higgins/ and the DigitalMe selector at http://www.bandit-project.org/index.php/Digital_Me

Furthermore the token is signed by the identity provider providing proof of authenticity of the originator (the identity provider) and integrity of the token itself which can be verified by the relying party.

Since the user has to concur with the identity provider, if the security token is encrypted for the relying party or in a token format unknown to the card selector, a so-called *display token* is also sent to the client. The display token contains the same claims as the security token, except that the contents of the display token are encrypted with the user's public key rather than the relying party's public key, so that the user can inspect the values filled in for each field. If the user concurs with the identity provider that the supplied claims are correct the user's client forwards the security token to the relying party.

4 Combining ePassports and User-Centric Identity

This section describes how the scenarios in Section 3 change when Information Card is combined with *e*Passports.

In the altered scenarios for enrollment and authentication we have the traditional three parties involved in user-centric identity management as described in Section 3, namely the relying party, identity provider and the user's client. These parties are depicted in Figure 1.

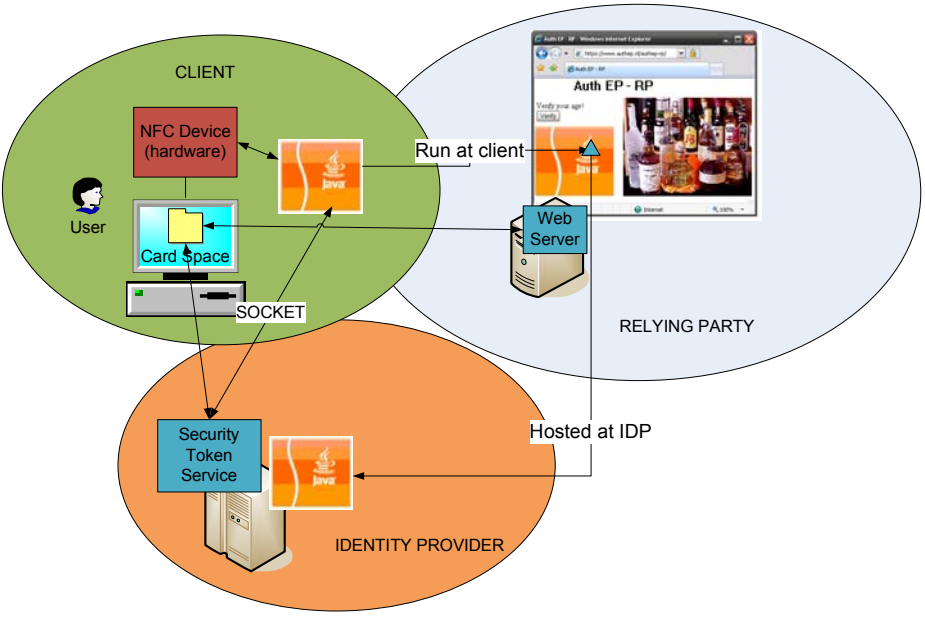

Fig. 1. The three parties in the *e*Passport Information Card scenario

4.1 Enrolling the ePassport at the Identity Provider

Before the user can use his *e*Passport at a relying party, he needs to enroll it with the identity provider. The user visits the identity provider's website and requests a managed card. The managed card will be tied to the user's *e*Passport. The user also supplies the BAC keys to the identity provider at this point. The identity provider needs the BAC keys in order to communicate with the *e*Passport chip during the authentication scenario as described in 4.2. Remember that the BAC keys are based on the user's date of birth, the *e*Passport's date of expiry, and the *e*Passport's document number.

At enrollment time the user sends an empty self-issued card to the identity provider which is used to back the managed card. The user also enters the date of birth, date of expiry, and the document number at the identity provider's website. The identity provider stores this information and sends a managed card (whose picture resembles a passport) to the user's card selector. The managed card contains no information apart from an id which the identity provider can use to later resolve the user's BAC keys (which it needs to communicate with the *e*Passport) and authenticate the user.

At enrollment time the user also needs to install a so-called Java policy file, allowing signed mobile code coming from the identity provider's web server to access the contactless card reader hardware. The role of the policy file is explained in Section 4.2. As an alternative to using this Java mobile code approach the user could be asked to install some local software, which might be perceived as being more transparent from the average user's perspective.

4.2 Using the *e*Passport to Authenticate at a Relying Party

Figure 2 shows the different entities involved in the authenticate-with-*e*Passport scenario and the traffic that is exchanged between them.

A TCP connection from the identity provider to the user's contactless card reader is created as soon as the user loads the relying party's login page. In the current implementation this is accomplished by placing a Java applet owned by the identity provider on the relying party's web page (to be more precise, what is placed on the relying party's web page is an HTML applet tag linking to applet code on the identity provider's web server). The applet is signed by the identity provider and also loaded from the identity provider's web server so that the Java Runtime Environment (JRE) at the user's client trusts this piece of mobile code enough to allow it to set up a connection back to the identity provider's server. The JRE was given permission to connect to the contactless card reader in a Java policy file which was installed during enrollment. The TCP connection is used for subsequent communication between the identity provider and the user's *e*Passport.

Using the managed card acquired during enrollment the user can attempt to login at the relying party. An information card policy is sent to the identity provider via the Card Selector just like in the normal Information Card scenario. One extra step is taken by the identity provider after receiving a token request from the client. In this extra step the identity provider checks if the user has a valid passport and it reads the user's details from the passport. As soon as the client actually requests a token at the identity provider, the identity provider will look at the provided managed card and

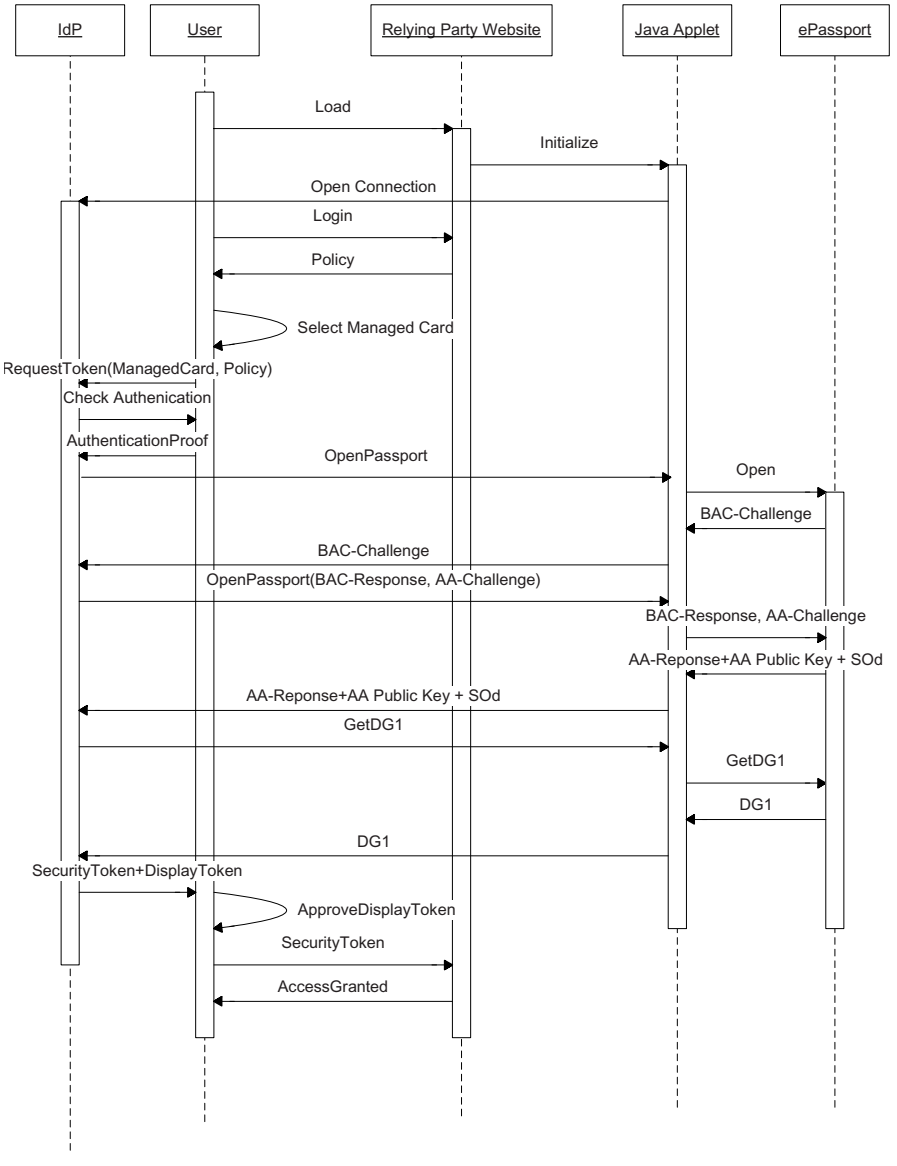

Fig. 2. Message sequence chart of the authenticate-with-*e*Passport scenario

send the appropriate BAC data to the passport authenticating the identity provider at the passport. The identity provider will request the *e*Passport's AA public key and SOd. With the SOd it can check if the public key has been signed by the issuing country. It can then send a random challenge to the *e*Passport which encrypts it using the AA private key. This proves that the passport is authentic and not a simple clone. The identity provider will request the minimal needed information from the *e*Passport to

confirm to the token request. The token is sent back to the client and from here on the normal Information Card scenario continues.

To summarize, the identity provider uses BAC, AA, and PA and then reads DG1. Based on the results of the security protocols the identity provider knows that the information in DG1 correctly identifies a citizen of the issuing country (for as far as the identity provider trusts the country's CSC, of course). Remember that DG1 contains basic textual card holder information (name, date of birth, date of expiry of document, document number, gender, nationality, and in the Dutch case even the citizen number). The information in this data group is used in the token created by the identity provider and only the required fields (as requested by the relying party's policy) are sent to the relying party (via the user's client). No other information is sent to the relying party and the relying party needs to trust the identity provider that it has done its job in checking the validity of the user's *e*Passport.

5 Discussion of Lessons Learned

The previous sections of this paper report on an experiment in which *e*Passports are combined with the user-centric identity management framework Information Card. It validates that it is in principle possible to create an identity solution combining real smart cards and 'virtual' information cards. This section discusses the lessons learned.

5.1 The Need for an Online Identity Provider

A surprising aspect of the described solution is that the role of the identity provider is somewhat different from traditional Information Card identity providers. A typical identity provider has knowledge about identities of users. The identity provider in our experiments, on the other hand, stores no information about the user except for the BAC access keys. The token sent to the relying party is freshly constructed based on information read 'live' from the *e*Passport, not on information stored at the identity provider. In fact, one could argue that the government issuing the *e*Passport should be considered as the real provider of identity here and that no separate identity provider should be needed.

Disadvantages of having a separate identity provider lie in the trust that the other parties need to have on the identity provider's integrity. These are discussed in Section 5.2. Still, this identity provider is necessary to deal with limitations in Information Card and ICAO specifications:

1. The Information Card specification simply requires an external online identity provider. One could, of course, always choose to implement it as a client-side or relying party-side service.
2. The ICAO specifications were never explicitly designed to allow limited disclosure of data in the sense of Cameron's second law. The information in the *e*Passport is structured in data groups and the complete data group needs to be sent to an inspection system before it can determine whether its hash in the SOd checks out. The online identity provider is needed to act as a privacy filter.

3. The identity provider stores the BAC keys and uses these to open a secure connection to the *e*Passport. Other parties - on the Internet but also client-side malware - cannot interfere with the communications between identity provider and *e*Passport[7]. This means that the client is not a part of the Trusted Computing Base. The BAC keys should not be distributed to untrusted parties for security and privacy reasons, providing them to each relying parties is therefore not an option.

A smartcard solution that would be designed specifically to facilitate privacy-sensitive assertions on identity claims, would need to be able to sign not only specific parts of the identity data, but also derived data from these identity data. For example, sign an 'above 18' claim, derived from the birth date.

5.2 The Need to Trust the Identity Provider

Since user-centric identity management introduces an online identity provider, a question that will need to be answered is to what degree both the user and the relying party must *trust* the identity provider:

- The user needs to trust the identity provider, which has full access to his *e*Passport, with respect to the privacy sensitive data stored in the chip and with respect to the authentication functionality provided by the chip. Obviously, our implementation of the identity provider is well-behaved with respect to privacy: it will only read relevant data groups and only use this to construct a token which is sent (via the user's card selector, i.e. with the user's consent) to the relying party. Still, the user needs to trust that our identity provider is implemented as advertised.
- The relying party needs to trust the identity provider with respect to the correctness of information issued about the user. The relying party needs to trust that the identity provider has done its job in inspecting that the *e*Passport is authentic (using PA) and is present during the transaction (using AA). The identity provider is not able to convey evidence (such as a signature) that proves to the relying party that the information in the token originates from an *e*Passport without sending the complete data group DG1.

A particularly frightening threat to the user's privacy would be an evil identity provider that keeps track of all relying parties visited by a user. Such a big-brother identity provider could even use the *e*Passport's active authentication functionality to construct undeniable proof-objects of the user's involvement in transactions with relying parties (i.e. have the user's *e*Passport 'sign' the transaction so as to later confront the user with such proof). Note, however, that there is no need for the identity provider to know the relying party's identity, it only needs a public key from the relying party to encrypt the security token.

Another threat is formed by the simple challenge-response nature of the AA protocol. A rogue identity provider can relay challenges and responses to another identity provider and pretend to have direct access to an *e*Passport.

---

[7] Unfortunately, client-side malware can communicate with the *e*Passport if it manages to get hold of the user's date of birth and the document's number and date of expiry through some other (totally unrelated) means.

How an identity provider can build up enough trust so that both users and relying parties trust it is a general trust problem. The following three suggestions are merely recommendations:

- The server should be run by an independent party. Although the government can be trusted in terms of correctness of information (i.e. the relying party would be happy with the government as online identity provider), the general sentiment amongst end-users may be that governments should not be present during each and every online transaction of their citizens. Governments may not be too eager to become online identity providers for their citizens either.
- The implementation of the identity provider should be open and transparent. Open source helps. However, it remains impossible to validate online whether a server is actually running the source code that is published.
- Trust in the identity provider could perhaps be established using *reputation*. Reputation based Trust (occasionally referred to as web-of-trust) is a mechanism that is sometimes part of Web 2.0 applications. *CACert.org* is an example of a real-world web-of-trust: it establishes a distributed network of trust by having people meet *in real life* and present their (non-electronic, old fashioned) passport in order to gain trust points. Persons with enough trust points can get their SSL site certificate signed by the CACert certificate authority. Web users can add the CACert certificate authority's root certificate in their browser's list of trusted certificates if they feel that the described procedure warrants that the CACert certificate authority only signs site certificates when the site's identity has been adequately checked.

The privacy problems with the identity provider are beyond the scope of the feasibility study described in this paper. A solution lies either in building up enough trust in the identity provider or in changing the *e*Passport standards so that it can be used to generate trustworthy proof of authenticity and at the same time conform to Cameron's second law. The latter option makes an online identity provider redundant. Recent interest in integrating different hardware electronic identity cards[8] (*e*ID), such as described in [3], indicates that we may be heading towards a world where online identity providers are no longer necessary or at least play a different role.

5.3 Not a Global PKI for Online User Authentication

Unfortunately our solution does not provide a global PKI for online user authentication. Apart from practical problems (not every citizen has an *e*Passport, contactless card readers are not widely available yet), the ICAO standards leave countries plenty of options in *not* implementing the various features which are essential to the proposed identity solution. The features that have to be present to allow online verification of an *e*Passport are:

- The *e*Passport must support passive authentication so that the authenticity and integrity of the LDS can be verified.

[8] See also the Stork European project at http://www.eid-stork.eu/

- The *e*Passport must support active authentication so that the authenticity of the chip can be verified.
- The data groups with identifying information must not be protected by extended access control (unless relevant keys are know to the identity provider).
- The document signing public key certificate must be available to the Identity Provider and its validity should be checked.

Passive authentication is present in all passports. Active authentication, on the other hand, is optional and recent interoperability tests[9] seem to indicate that less than thirty percent of countries currently chose to implement active authentication.

The data groups of interest for this paper, DG1 and DG15, are not protected by extended access control: they are readable to an inspection system once the basic access control protocol is successfully completed. Future versions of the *e*Passport may contain interesting identifying information in other data groups that are protected by extended access control. In that case the identity provider needs to be trusted by the ICAO Public Key Directory (PKD) to get access keys for performing the extended access control protocol.

Some countries have their country signing certificate available on a public government website. An overview list appeared in an ICAO published report [2]. As an alternative to publishing this certificate on a website the PKD can be used. The PKD shares document signing certificates rather than country signing certificates. An additional advantage of the PKD is that it provides a central online service for obtaining validity information about certificates in the form of so-called certificate revocations lists.

Our solution was tested using the Dutch *e*Passport which supports all features necessary for our purposes. The ICAO standards for *e*Passports are definitely not designed with online verification and limited disclosure of contents in mind.

6 Concluding Remarks

The main conclusion from this research is that with the help of an online identity provider it is possible to leverage *e*Passports for user-centric identity management. Our prototype validates this for Information Card, and we expect a similar outcome for e.g. OpenID, or for a more identity provider-centric approach to identity management such as SAML.

The online identity provider needs to be trusted by both user and relying party, but due to specifics of the Information Card specification and the *e*Passport specification, cannot be done without.

From a privacy perspective, and to enhance availability, it may be preferred to have an identity solution which does not depend on an online identity provider at all. After all, the *e*Passport can be considered as a smartcard that embeds several important claims about a person (such as name, birthday, gender), which are already signed by a trusted government agency. To be useful in a privacy sensitive manner, the *e*Passport or another similar hardware token with embedded claims would have to be able to sign individual claims, including derived claims such as ‘above 18”.

[9] See http://www.e-passports2008.org/

Apart from further exploring the above, there are some technical loose ends that require our attention for future work:

- OpenID is the other popular upcoming user-centric identity management standard. It would certainly be interesting to investigate how our solution integrates with OpenID.
- Currently the managed information card associated with an *e*Passport is backed by a self-issued information card of the user. A much more elegant alternative would be to use the smart card framework supported by the card selector (the Crypto API for Microsoft Windows or PKCS 11 for certain other alternatives) instead. In essence this creates a "soft-token" interface for the *e*Passport's active authentication signature which can be used to back a managed card. Such an approach of integrating smart cards with Information Card would also be more in line with [1] and commercial smart card based identity solutions such as TrustBearer's OpenID product[10].
- The introduction of Near Field Communication (NFC) technology in all sorts of mobile devices may solve the practical problem of (lack of) contactless card reader availability. While porting *e*Passport software to J2ME may be an interesting challenge, the eCL0WN tool[11] for Nokia NFC phones proves that it is feasible to read out *e*Passports using NFC. Similarly, IBM is working on using contactless cards to strengthen online authentication [12]. Combined with user-centric identity management systems for mobile phones (IBM has demonstrated a card selector for the Android platform [11]) this leads the way to *mobile-centric identity management*, which combines user-centric with mobile phones as the most ubiquitous and personal device people have. The *e*Passport could, for example, be helpful in the provisioning process in such a mobile-centric identity management system.

Acknowledgments. This research was funded by the NLnet (http://nlnet.nl/) foundation.

References

1. Aussel, J.-D.: Smart Cards and Digital Identity. Teletronikk 3/4, 66–78 (2007) ISSN 0085-7130
2. Broekhaar, S., Verschuren, J.: How to Obtain CSCA Certificates – The CSCA Overview List, MRTD report, 2, ICAO, 32–35 (2007)
3. Bruegger, B.P., Hühnlein, D., Kreutzer, M.: Towards global eID-Interoperability. In: BIOSIG 2007. LNI, vol. 108, pp. 127–140 (2007)
4. Cameron, K.: The Laws of Identity – as of 5/12/2005, Microsoft Corporation (2005)
5. Hoepman, J.-H., Hubbers, E., Jacobs, B., Oostdijk, M., Schreur, R.W.: Crossing Borders: Security and Privacy Issues of the European e-Passport. In: Yoshiura, H., Sakurai, K., Rannenberg, K., Murayama, Y., Kawamura, S.-i. (eds.) IWSEC 2006. LNCS, vol. 4266, pp. 152–167. Springer, Heidelberg (2006)

[10] See http://www.trustbearer.com/
[11] See http://seclists.org/fulldisclosure/2008/Dec/0567.html, Jeroen van Beek.

6. ICAO: Machine Readable Travel Documents, ICAO Doc 9303, part 1: Specifications for Electronically Enabled Passports with Biometric Identification Capability, 6th edn., vol. 2 (2006)
7. ISO: Information technology — Security techniques — Digital signature schemes giving message recovery — Part 2: Integer factorization based mechanisms, ISO/IEC 9796-2, 2nd edn. (2002)
8. Juels, A., Molnar, D., Wagner, D.: Security and Privacy Issues in E-passports. In: Proc. SecureComm 2005, pp. 74–88. IEEE Computer Society, Los Alamitos (2005)
9. Lekkas, D., Gritzalis, D.: e-Passports as a means towards the first world-wide Public Key Infrastructure. In: López, J., Samarati, P., Ferrer, J.L. (eds.) EuroPKI 2007. LNCS, vol. 4582, pp. 34–48. Springer, Heidelberg (2007)
10. Liu, Y., Kasper, T., Lemke-Rust, K., Paar, C.: E-Passport - Cracking Basic Access Control Keys. In: Meersman, R., Tari, Z. (eds.) OTM 2007, Part II. LNCS, vol. 4804, pp. 1531–1547. Springer, Heidelberg (2007)
11. Nadalin, A.J.: Mobile Identity. In: The European e-Identity Conference, The Hague (2008), http://www.eema.org/downloads/annual08/nadalin2c.pdf
12. Nanda, A.: Identity Selector Interoperability Profile, V1.0, Microsoft Corporation (2007)
13. OpenID: OpenID Authentication 2.0 – Final (2007), http://openid.net/specs/openid-authentication-2_0.html
14. Ortiz-Yepes, D.A.: Enhancing Authentication in eBanking with NFC-Enabled Mobile Phones. ERCIM News 76, 63–64 (2009)
15. SAML, OASIS specification (2005), http://saml.xml.org/saml-specifications
16. Vaudenay, S., Monnerat, J., Vuagnoux, M.: About Machine-Readable Travel Documents. In: Proc. International Conference on RFID Security 2007, pp. 15–28 (2007)
17. Vaudenay, S.: E-Passport Threats. IEEE Security & Privacy, 72–75 (November/December 2007)