Personas: Beyond Identity Protection by Information Control

A Report to the Privacy Commissioner of Canada

D.B. Skillicorn and M. Hussain

March 2009

Updated April 2009

Abstract

As individuals interact on larger and larger scales, what makes up their identity also has to expand. In a village, only a single name is enough. In a country, other attributes such as a social insurance or taxpayer number become necessary. Across the world, a multinational hotel chain may want to identify the same individual whenever he or she stays at one of their hotels, but there is no single global identifier that makes this possible.

Because identities are important in so many interactions, not all of them known to and supervised by individuals, there are considerable economic and privacy risks when identity information is misused. Today, the standard solution to this problem is to mandate legal or policy rules that restrict the flow and use of identity information.

This solution is starting to fail for two reasons. First, and most importantly, new developments in data mining and data fusion allow identities to be constructed from data that has not previously been considered identifying. Systems often do not control this kind of data as tightly as traditionally identifying data. Second, increasingly information that was supposed to be controlled is released accidentally. Once this has been done, there is no way to call it back, but also no way (short of Witness Protection Programs) to create fresh identities for individuals who have been compromised.

We design and describe a new approach to this problem, based on the creation and use of artificial identities, called *personas*. These identities can only be traced to the individuals who created them under tightly controlled circumstances, but personas can act for individuals in almost all situations, from ecommerce to border crossing.

1 Identities

The concept of *identity* is a complex one and has been viewed from many perspectives. For example, the question of why the two-year old, twenty-year old, and sixty-year old versions of the same individual feel as if they are the same person led to the study of consciousness as the defining aspect of individual identity. In what follows we will restrict ourselves to a more pragmatic definition of identity [1]: the set of attribute values that uniquely label a single individual in some specific context. For example, in a village, an individual's name may be their identity; in a country, this is probably not enough, but a Social Insurance Number may serve as an identity. Different contexts have used different forms of identities: governments often use identifiers such as social insurance and taxpayer numbers; banks and merchants use account numbers, credit cards numbers, or customer loyalty card numbers; and families use nicknames to distinguish family members whose official names may be the same.

The attributes that are used to form an identity fall into three categories:

1. Attributes associated with physical and mental existence as an individual. These attributes include biometrics such as fingerprints, iris patterns, and voice qualities, but also skills that are hard to learn and transfer such as fluency in a particular language, or discriminating taste in wine (used in a Dorothy Sayers short story). Such attributes are attractive as ways to construct identity because they are not readily transferable or forgeable (although many biometric systems are actually much easier to spoof than they seem).

2. Attributes authenticated by a trusted other party, often a government. Examples include name (authenticated by reference to a birth certificate or birth registration) or citizenship. Such attributes often carry with them a set of *rights*, which is why it is worth the authentication agency's effort to maintain the data necessary to provide authentication.

3. Attributes chosen by the individual to act as (part of) their identity, for example a nickname or screen name.

These attributes have the characteristic property that they discriminate well between individuals in a given context – knowing the value of even a single one will often be enough to select the unique individual it belongs to.

When individuals participate in transactions, such as going to a bricks-and-mortar store or an online one, they also use other attributes that are not usually thought of as part of their identity. For example, to buy something they must prove to the merchant that they will (eventually) pay for what is purchased. So they must provide values for attributes such as bank balance, credit rating, or credit card number. Similarly, when an individual applies for a job, they may need to show that they have obtained university degrees, or do not have

a criminal record. They must also indicate their desires: what they want to buy, or which job they want to apply for.

This second class of attribute values are not usually thought of as part of an identity – after all, many people buy the same things, bank balances fluctuate from week to week, and many people might apply for the same job. Nevertheless, recall the definition of identity; given values for only a few of these attributes may be enough to uniquely identify each individual, and so they become identifying. The development of data-mining and data-fusion tools makes it much more practical to actually make identifications from the values of such attributes, and it is often in the interests of, for example, merchants to do so.

This has a major impact on privacy because it means that it is no longer possible to be involved in many kinds of transactions without being identified. *All* attributes have become potentially part of identity. Conventional approaches to maintaining privacy do so by controlling (restricting) the flow of some attributes (the "identifying" attributes) but not of others. However, solutions like this can no longer work when all attributes are potentially identifying.

2 The problem with controlling identity by controlling information flow

Most approaches that allow individuals to control their privacy and how their personal information is used rely, in some way, on controlling the flow of identity information. For example, legislation and many business's privacy policies typically state how information that is collected will be used, where it will be stored, who else will be allowed access to it, and for what purposes. Many businesses place such information in data warehouses where it will be analysed to build models of customers using data-mining technology. There is also often an implicit assumption that if the data can anonymised or de-identified (that is, if the identity information can be removed) then the data can be used more widely and perhaps even given to others. Of course, we argue that anonymisation or de-identification cannot be properly done, since the values of any remaining attributes may still form identities. For example, data such as a zip code appears to be an attribute associated with an address. However, together with another attribute such as the fact that a family has six children, it may be enough to identify that family. Results from the analysis of collaborative filtering data suggest that individuals can be identified, with high probability, knowing only their taste for two or three less-popular items

Of course, in some settings it may be possible to create identities from data, but not to map these identities to the more conventional ones, involving names and addresses. This may sometimes be acceptable, but it is risky because public information, unknown to the parties doing the modelling, may enable the mapping regardless. This happened when AOL

released a list of the searches carried out by its users *anonymously*, that is with the identifiers removed. Some users were still identified because they had searched for themselves; others were identified by friends who guessed what they might have searched for. And many people will have had the experience of hearing someone described without their name, and realizing that they know who the person (probably) is, just from their characteristics.

To further illustrate the issues, we consider a number of identity and transaction scenarios.

Scenario 0: Bob goes to his corner store several times a week and buys something using cash.

Paying for a transaction using cash requires no identifying information; the cash itself is authenticated by the government for use by anyone ('bearer'). However, note that Bob's appearance does provide the shopkeeper with some weak identity information. For example, if Bob looks scruffy, he may receive poor service from the shopkeeper. If the police were to question the shopkeeper about the transaction, he would be able to provide a description that might become a partial identity for Bob.

Since Bob goes to the corner store repeatedly, the shopkeeper is able to build a model of Bob, even though Bob always uses cash. His weakly-identifying appearance attributes allow the shopkeeper to know that each visit is by the same person. This may be exploited to provide Bob with differential service or differential pricing. This illustrates what is called the multishow linkability problem: repeated transactions allow an identity to be constructed based on whatever attributes are available (even those that are not usually considered identifying), and on the objects purchased. For example, if Bob rents DVDs from the corner store, the shopkeeper can build a model of Bob's viewing tastes as well as his appearance. Eventually his viewing tastes could become an identity.

Scenario 1: Zorfan is travelling in a foreign country and wants to pay for something using a travellers cheque. The shopkeeper insists that he shows his passport before he will accept the cheque. When he discovers Zorfan's nationality, he becomes abusive.

This scenario shows how many transactions demand identity information that is not actually needed. Travellers cheques were designed to be self-validating; a user signed them once when they were first received, and then signed them again when they were cashed to prove that they were the person who received them. This proved inadequate, so that cashing travellers cheques now typically requires showing some sort of government-authenticated identity document. There is a vicious cycle here: the weakness of identity management systems requires exposing more identity information which in turn increases the probability that this identity information will leak, and further weaken identity management. And any robust identifier tends to become required in circumstances for which it was not intended, binding, for example, government and commercial identities. Part of the reason that identity theft is a bigger problem in the U.S. than in Canada is the American Social Security numbers were used as unique identifiers in many different settings, some of which are excluded by law

in Canada.

Scenario 2: Sarah makes multiple visits to Amazon to buy books. Because she signs in each time, Amazon knows that these purchases have been made by the same person, and can build a record of all of them. From this history, Amazon derives a model of her tastes and begins to make personalized recommendations to her about possible future purchases.

Transactions at a distance look, at first glance, as if they should be more anonymous than Bob's purchases at the corner store, but there is a need for stronger authentication that tends to require *more* identity information.

Although the record of the purchases Sarah has made do not need to be associated with her name to execute the recommendation algorithm, if Amazon revealed the complete list of purchases, it is possible that Sarah's record could be associated back to her.

Sarah has no choice but to reveal her identity because Amazon has to know that it will be paid. Even if Sarah uses a one-time credit card number (a new and rarely used technology), she must still provide an address for delivery. Although an address is just an attribute, it is also strongly identifying since relatively few people live or work at any particular address.

Scenario 3: To prevent any one retailer from building a complete model of his shopping habits, Mike buys some of his books at Indigo, and some at Chapters. Mike does not realise that these two chains are actually the same organisation underneath, that they share information about their customers, and so are able to build the complete model he was trying to prevent.

Of course, if Mike pays with a credit card, then the credit card company can put together all of the shopping for which he uses it, so that organisations like this, whose role has traditionally been to guarantee that businesses will be paid, can also assume the role of user modeller. (They already build such models, but for the purpose of detecting fraud.)

This scenario illustrates the increasingly common situation where different organisations share data, unknown to the individuals concerned. In this case, even PIPEDA might not provide protection; it might require very astute consumers to realise that they were agreeing to much wider data sharing than it seemed.

These scenarios illustrate some of the problems of trying to control identity and preserve individual privacy by controlling information flow. When every transaction reveals a large part of an individual's identity, often unnecessarily, and counterparties have strong incentives to model individuals, a considerable amount of privacy is surrendered. And this happens even when things go right. When there are problems, and information is leaked improperly, an individual's privacy can be completely destroyed. The rise of identity theft, a crime now as significant in revenue terms as drugs, shows how easily enough data can be obtained to simulate someone else's identity in commercial and government settings.

If *all* attributes are, at least to some extent, identifying, then solutions that rely on controlling or limiting access to, or transfer, of identity attributes but allow other attributes to be shared or moved freely cannot provide protection for identity. Even in the best case, such solutions put the power over identity information in the hands of someone or some organization other than the person to whom it belongs.

If controlling information flow cannot protect identities, are there other solutions that can? We suggest that there are, and design one in the following sections.

3 Artificial identities – personas

The solution to preserving identities, and keeping control of them in the hands of those they belong to, is to use *personas*, artificial identities with the following properties:

- Personas are unique, so that any persona belongs to exactly one real individual.
- An individual can generate many personas for use in different, or even the same, contexts. The fact that these personas belong to the same real individual cannot be deduced from seeing any or all of them.
- Personas cannot be linked to the individual to which they belong, except by using a well-defined and protected process involving an intermediary. This allows, for example, law enforcement to track a persona to its owner when a crime has taken place. Hence personas cannot be repudiated.
- Personas can include attributes about the properties, rights, and desires of the individual they represent, and these can be authenticated if required.
- The associated attributes of a persona are packaged as tokens that allow other entities to prove to themselves that the persona has the attributes, rather than as values. For example, to demonstrate that an individual is old enough to vote, a persona does not have an attribute field whose value is the individual's age but rather carries a proof that the individual's age is greater than 17.

Personas solve the data fusion problem. An individual can create and use personas whenever they are needed. These personas can be used in place of the individual in any setting. The other parties in these settings see each persona as different in a strong sense – they cannot be identified as belonging to the same individual.

Rudimentary forms of personas already exist. In a sense, cash is a kind of persona. Someone who buys something with cash does not have to reveal his identity; instead the cash is guaranteed by an intermediary, the national government. There are also mechanisms

to create email addresses without having to identify oneself, and these are often used when an email address is needed but a user is concerned about making their main email address known to someone they do not necessarily trust.

We now turn to the mechanisms that enable personas to be created and used.

4 Creating and managing personas

There are four components to the system of creating and using personas:

1. Individuals who want to use personas;
2. Persona providers (PP) who generate personas, and act as guarantors that each persona belongs to a real person, and has the properties claimed by that person.
3. Service providers (SP) who interact with personas to provide services.
4. De-anonymization authorities (DA) who trace personas, with the help of PP, to individuals. An PP can play the roles of both an PP and a DA.

To illustrate how these components are used, we first go through the simplest example, in which an individual wants to buy something from a store without revealing his real identity.

The individual first goes to the identity provider with his real identity and any relevant attributes that they want to be associated with this persona. (We defer for the moment the question of how the identity provider knows that the individual is entitled to these attributes and values.) In the context of shopping, an attribute would typically be the amount of credit associated with the persona.

The identity provider takes this information and returns a persona to the individual who asked for it, keeping a record of the association of individual and persona in case it is ever needed for de-anonymization. A persona can be thought of as an essential identity, plus a list of attributes that have been guaranteed by the identity provider, packaged in a way that prevents the owner from changing or unpacking the list. The attributes are not data values, but standalone proofs that the persona possesses those attributes.

The individual can now use the persona to generate any number of *locked personas*. Each of these locked personas looks different and cannot be associated either with the individual or with each other. A locked persona plays the role of the individual in a particular transaction. (It could be reused in multiple transactions but this defeats its purpose, since a locked persona always looks the same and so can be fused across transactions.)

An individual might want to use different parts of the list of attributes with different service providers or different transactions. This could be done by asking the identity provider to provide multiple personas, each with a different subset of the list, but this is a clumsy solution. Instead, the locking process can select some of the attributes from the list to be accessible by the service provider, while others are completely blocked from access.

The locking process can incorporate information from the intended target service provider, if desired. This prevents the locked persona from being used anywhere else.

The locked persona is then sent to the service provider along with instructions about what is being purchased. The service provider can determine, from the locked persona, whether or not the persona has enough resources to pay for the purchased items. The service provider does not have to consult with the identity provider to complete the transaction. This is of critical importance, because identity providers cannot guarantee to be constantly available. If they were required to be, they would be a serious point of failure for the whole system. This also means that a persona can be used in a completely offline way, for example by placing it on a smart card which could be checked by standalone devices, for example to control access to buildings.

Figure 1 shows the flow of personas among the different components. A persona is generated at an PP and sent to an individual. The individual generates a locked persona and sends it to an SP. The SP verifies the locked persona in a completely standalone way. If there is a problem with the transaction, for example the individual is trying to rob the merchant in some way, the SP can send the persona to a DA. The DA extracts an identifier, and sends it to the PP. The PP is the only component that can reconstruct the mapping of the persona back to the individual.

4.1 Persona Providers, PP

An individual contacts an PP and claims a set of attributes. The PP validates the individual claims and generates the required persona. The process of validating the individual claim is highly dependent on several issues, for example, PP policies and the type of attributes claimed. Normally, this process is considered out of the scope in the majority of the work in this area. Take NetFile as an analogy. NetFile is a Canada Revenue Agency web service that allows taxpayers to electronically submit their tax returns. NetFile is only accessible with a personalized access code. CRA provides eligible taxpayers with the needed personalized access code to use NetFile. Determining which taxpayers are eligible for an access code, however, is not part of NetFile.

There are three scenarios that may occur at the PP when it receives a persona request.

1. The individual's claim is directly verifiable by the PP. An individual who has a bank

account at BMO may login to his account and ask BMO for a persona that can be used for online shopping. BMO then checks the individual's balance and provides the required persona. BMO associates the newly generated persona with the individual's account.

2. The claim is not directly verifiable by the PP, but the claim is backed up by another PP. The individual may provide a persona from another PP with attributes that back up his claim. For example, MasterCard may allow individuals with personas from BMO to obtain a credit card. MasterCard associates the newly generated persona and the individual's BMO persona.

3. The claim is not verifiable by the PP. In this case the PP may require the individual to be present or to call an agent, etc.

This is not very different from how authentication works in the real world. Almost all attributes are derived from a few basic identity attributes, usually authenticated by governments: to buy something I use a credit card, which is backed up by my credit rating, which is backed up by my bank account balance, which required a government identification to open.

Once the PP verifies the claims (the desired attributes) of the individual, a persona is generated as follows.

- The individual's claims and identity are wrapped together and digitally encoded.
- The encoded package is digitally signed by the PP.
- The digitally signed package is sent back to the individual.

The full technical details of generating a persona are provided in Appendix A. Identity providers in our society are of a number of different types. Governments guarantee personal identities such as citizenship; schools and universities guarantee credentials such as degrees; supervisors guarantee character or performance by writing references; and financial institutions guarantee financial worth.

4.2 Individuals

Each individual can receive personas from multiple different PPs as they wish, and can store these personas for later use. For example, the individual in Figure 2 receives personas from three PPs. The individual follow these steps to use a persona.

- (Optional) The individual checks an SP site that they wish to use for specific information that the site requires of personas.

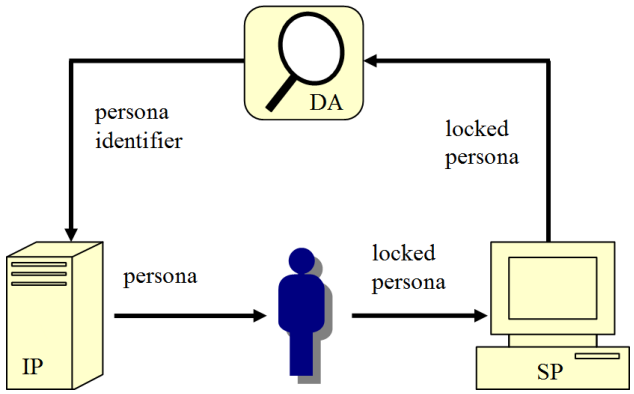


Figure 1: The entities generating and using personas

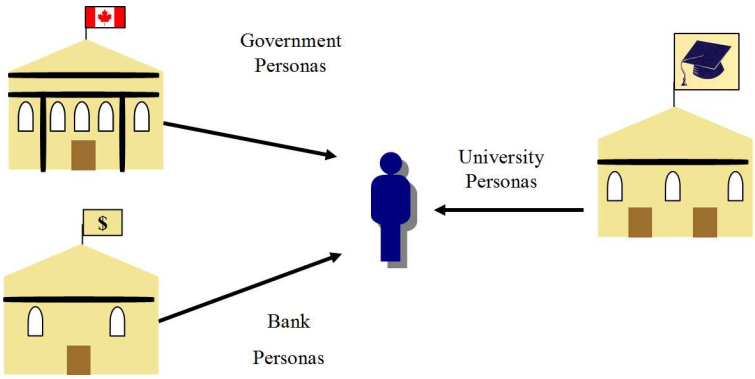


Figure 2: PPs providing an individual with personas

- The individual generates a locked persona from one or more of the required personas, selecting which of the attributes associated with the persona should be visible in the locked persona.
- The individual sends the locked personas to the SP, which then provides the service, confident that there is a real and entitled person behind the transaction, but unable to deduce who that person is.

The full technical details of using a persona are provided in Appendix A.

Figure 3 shows the same individual using many services with her personas. The government persona is used to prove that she is over 18. She also enjoys the student offer at her phone company after providing a locked persona obtained from her university that guarantees she is eligible. She downloads music from an online store with a locked persona obtained from her bank.

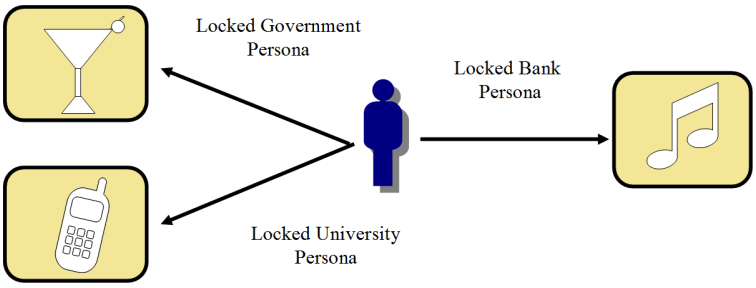


Figure 3: An individual using her personas at various SPs

4.3 Service Providers, SP

Service providers can be any web service on the internet, or indeed any real-world service provider, subject to the caveats of Scenario 0. Each SP keeps a list of PPs that it trusts. Individuals with personas from the trusted PPs are allowed to use the services at the SP. For example, Amazon accepts payments made using Visa and Mastercard credit cards. Whenever an SP receives a request from an individual, the SP follows these steps.

- (optional) The SP tells the individual what properties of personas are required (Step 2 from Subsection 4.2).
- The SP receives the locked personas (Step 5 from Subsection 4.2).
- For each locked persona, the SP verifies, in a completely standalone way that does not require interaction with an PP, whether the locked persona is valid, including what visible attributes have been associated with it.

The full technical details of the process of verifying a persona are provided in Appendix A.

4.4 De-anonymization Authorities, DA

DAs are responsible for tracing locked personas to individuals, with the help of PPs. PPs and DAs can be implemented as separate components but might often be implemented within the same system. Figure 4 shows two scenarios involving DAs. One scenario comprises a bank, which has two components: an PP providing personas, and an DA tracing them. The online music store sends a locked bank persona to the bank in order to collect the money it is owed as the result of selling something to that persona. The bank traces the locked persona back to the individual who spent the money.

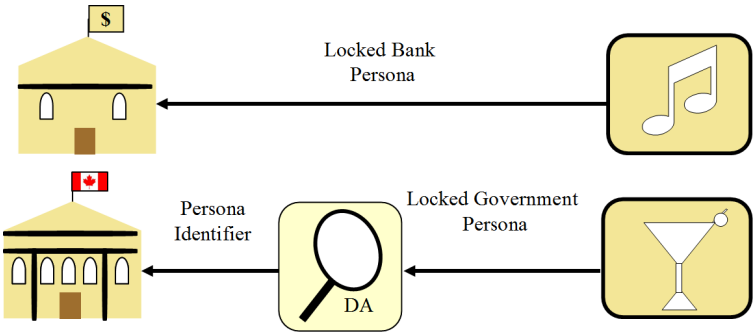


Figure 4: De-anonymizing personas using DAs

The other scenario comprises a wine store sending a locked persona to a DA. The DA extracts an identifier which allows the government to trace the locked persona to an individual.

Appendix B provides a detailed example of how the entire system can be used.

5 Other properties of personas

5.1 Relationship Linking

Relationship linking is an important feature that allows the encoding of relationships among personas. When the personas are presented to an SP, the SP may verify the relationships among these personas. Yet, all showings of these personas are un-linkable. That is, if these personas are presented again to the same SP, the SP cannot tell whether these personas have been presented before.

For example, a couple may have two personas linked with the '*couple*' relationship. When the two personas are presented simultaneously to an SP, the SP can verify that the two individuals behind the personas are a couple. If the couple visit the SP at a later time, the SP cannot distinguish this couple from other couples.

The same feature may encode more complex relationships, for example encoding relationships involving many individuals. The individuals involved in a more complex relationship can create related personas which they can present to an SP. The SP can verify the relationship among these individuals. If the individuals revisit the same SP, they cannot be distinguished from any other group of individuals having a relationship with the same structure.

Subsection A.4 of the appendix provides the technical details of this feature. Figure 5

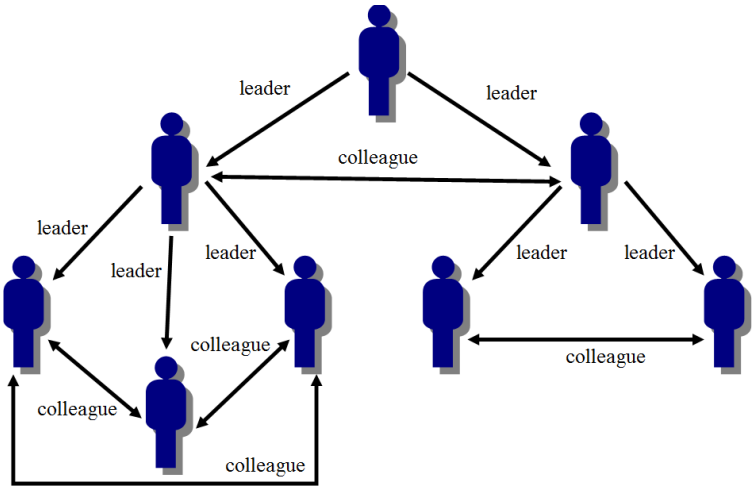


Figure 5: Tree-like relationship among project members

illustrates a scenario where the relationship among project members has a tree-like structure.

This feature can be used to require that a specified set of people in a specified relationship must all participate in an action at once. For example, using a business account can require the simultaneous action of several specified company officers. Allowing a child to cross a national border can be permitted only in the presence of at least one of her parents.

5.2 Nesting Personas

Personas can be nested, that is, an individual can receive a persona from one PP, generate a locked persona, and send it to another PP. The second PP treats the locked persona as an attribute and generates a new persona that contains the original locked persona. The process may be repeated at further PPs as well. Figure 6 demonstrates how personas can be nested. The advantage of nesting personas is that it allows individuals to combine personas into a larger one, while still allowing SPs to verify each of the component personas independently. The disadvantage is that a locked persona doesn't change. Once a locked persona is treated as an attribute and wrapped inside a new persona, any service provider that is allowed to access that attribute can match it with other service providers that can also access the attribute.

5.3 Resisting Replay Attacks

A replay attack occurs when someone copies the information transmitted between protocol participants, and then resends the information to create a second occurrence of a transaction.

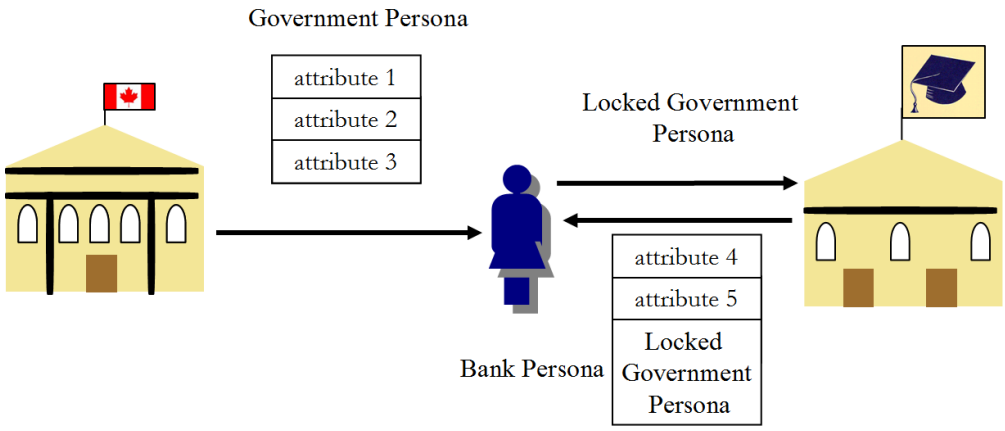


Figure 6: Nesting personas

For example, if an attacker is able to copy a credit-card number transmitted between an individual and an SP, the attacker may reuse the credit card number at the same or other SPs. Personas are not as susceptible to such attacks for two reasons. First, what it is transmitted between an individual and an SP is a locked persona dedicated to that SP, so it cannot be re-used at other SPs. Second, a locked persona can be stamped with a time-stamp, so that it can only be used for a short period of time. This helps an SP to detect locked personas that are copied and retransmitted.

Note that this ability to include contextual data as attributes of a locked persona has numerous benefits. For example, passport theft can be made pointless by having each traveller create a fresh persona-based passport for each trip, and including the border crossings and approximate times for which it will be used.

6 Related work

Several identity-management systems that incorporate some of the ideas we have been discussing have been proposed. Some well-known examples are Liberty Alliance [13], Shibboleth [16], WS-Federation [18], CardSpace [8], U-Prove [9], Idemix [6], and Prime [7]. We briefly survey them here.

6.1 Liberty Alliance, Shibboleth, WS-Federation, and CardSpace

Liberty Alliance is a consortium that includes Sun, HP, General Motors, and many other global corporations. The consortium's mission is to provide open standards for the development of federated identity-management systems. The architecture advocated by Liberty

Alliance has three frameworks: the Identity Federation Framework (IDFF), the Identity Web-Services Framework (IDWSF), and the Identity Services Interface Specifications (IDSIS). IDFF provides a single means for sign-on for individuals, and simple session management. IDWSF uses IDFF to establish permission-based attribute sharing, service discovery and description, and the associated security profiles. IDSIS builds on both IDFF and IDWSF to support a wide range of applications such as e-wallets and calendar services [15]. Shibboleth from Internet 2 is an open-source IMS for single sign-on across organizations. Shibboleth is used mainly by educational institutions.

Liberty Alliance and Shibboleth use SAML for specifying the communication of identity-related information between individuals and providers. SAML 2 extends SAML 1.1 to include the Liberty Alliance IDFF and Shibboleth. Therefore, Liberty Alliance 2 and Shibboleth 2 use SAML 2 as a basis for secure communication and for basic identity services such as single sign-on [15].

WS-federation is part of WS framework by Microsoft and IBM. WS-federation defines how identities can be federated among different providers. While Liberty Alliance uses SAML, WS-federation utilizes XML digital signatures standards. As with other federated IMS, WS-federation defines how identities can be federated among different providers.

CardSpace, from Microsoft, is also based on WS-*. CardSpace focuses maximizing individuals control over their identities. This is done by allowing the identities, called info cards, to be managed at the individuals' machines.

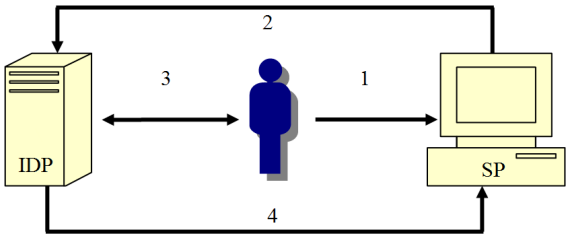
Figure 7(a) shows a typical usage scenario in Liberty Alliance, Shibboleth, WS-Federation and CardSpace. When an individual visits a service provider's (SP) site, the SP redirects the individual to her identity provider's (IDP) site. The IDP authenticates the individual and redirects her to the SP, where the redirection message contains a proof that the individual has been authenticated. If the individual navigates from the SP to another SP, within the same circle of trust, the individual may not be redirected again to her IDP. The SPs exchange the authentication information of the individual.

Some of the problems with these systems are:

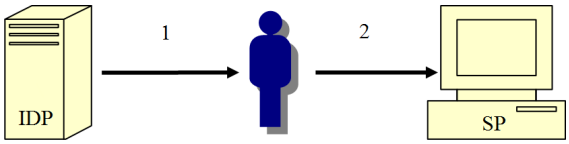
1. They allow 'non-identifying' information to be exposed to SPs. Using data-fusion techniques, this information can be merged to discover complete identities.

2. Federated identity-management systems employ privacy policies. It only takes one ill-designed policy to undermine the security and privacy of a whole system. The market is full of examples where the information from millions of credit cards was disclosed due to inappropriate decisions [2, 17].

3. Identity providers (IP) must be available whenever individuals interact with service providers (SP). This is an onerous quality-of-service requirement. Availability issues



(a) Federated identity management systems



(b) Anonymous certificate systems

Figure 7: Typical scenarios in IMS

themselves may provide metainformation that can help identify individuals. For example, an individual's country of origin may be inferred from the hours of the day when identity verification for that individual is available.

6.2 U-Prove, Idemix, and Prime

U-Prove, Idemix, and Prime are anonymous credential systems. These systems utilize zero-knowledge proofs to enable individuals convince SPs that those individuals are certified by IDPs, without disclosing any other information. Originally developed by Credentica, U-Prove has been recently acquired by Microsoft. U-Prove focuses on security and privacy aspects of identity management. Idemix, developed at IBM, not only targets privacy, but also tackles the problem individuals sharing their credentials. Prime is another anonymous-certificate system funded by the European Union and several corporations. European regulations regarding privacy are the core requirements of incorporated into Prime. Prime uses Idemix protocols for issuing and verifying credentials.

A typical usage scenario in U-Prove, Idemix Prime goes as follows (refer to Figure 7(b)). Whenever an individual desires to authenticate at an SP, the individual uses a certificate obtained previously from her IDP. The system uses zero-knowledge proofs to convince the SP that the individual has a valid certificate from the IDP.

As with the previous IMS, anonymous credential systems do not prevent attributes from being shown to SPs. This makes them susceptible to identity fusion and profiling. In addition to selective release of attributes and multishow unlinkability, this paper introduces an important feature, relationship verification. This feature helps individuals to prove relations between personas, while preserving multishow unlinkability. Implementing relationship ver-

ification in anonymous credential systems either violates multishow unlinkability or incurs major computation overhead.

7 Discussion and conclusions

We have discussed the challenges facing identity management approaches today, including the ease with which identities can be stolen and misused, and the extent to which new data-mining and data-fusion create new challenges because of their ability to derive identity from apparently innocuous attributes.

We have argued that solutions based on trying to control information flow, either by legislation or policy (such as businesses' privacy policies) are fragile because they would have to limit almost everything to be effective, and because accidental data release is easy and irretrievable.

Instead we suggest the use of artificial identities, personas, that can stand in for individuals in almost all circumstances, because they can be created with a full spectrum of properties, attributes, claims, desires, and relationships. Because these personas are, in a fundamental way, *single use* the potential for harm and loss of privacy are severely limited. The underlying properties are provable, and depend on strong cryptographic properties.

The overheads required to use personas are not trivial, but they are modest in most computer-supported environments.

A Technical mechanisms to support personas

This section presents the technical details that allow for personas to be generated, used, and traced.

A.1 P-IMS Operations

Operations at PPs

$setupAtPP : security\_param \rightarrow PPPukey \times PPPrkey$

setupAtPP generates the public-private key pair (*PPPukey* and *PPPrkey*) for the PP. Note that *setupAtDA* generates the public-private key pair for the DA.

$wrap : attributes \times locked\_persona \times PPPrkey \rightarrow persona \times secret$

wrap is executed by an individual D at a PP. *wrap* takes a set of claimed attributes and a locked persona (proof) that D is entitled to the attributes claimed. A persona P is returned which attests that D is entitled to the attributes. D uses the secret to prove the possession of P.

$check\_wrap : persona \times secret \times PPPukey \rightarrow boolean$

check_wrap is used by an individual or a PP to check if a persona is valid.

Operations by individuals

$show : persona \times secret \times DAPukey \times PPPukey \rightarrow locked\_persona$

Whenever an individual D wants to use a persona P at an SP, *show* is executed to generate a locked_persona L proving the ownership of P. D then sends F to the SP. Showing a persona can be associated with an action, message, time-stamp, *etc.* That is, we can treat *show* as a signature on a message or an action.

$selectiveShow : persona \times secret \times DAPukey \times PPPukey \rightarrow locked\_persona$

An individual D may use *selectiveShow* to show a subset of a persona P to an SP.

Operations at SPs

$verify : locked\_persona \times PPPukey \times DAPukey \rightarrow boolean$

An SP receiving a locked_persona L, uses *verify* to check whether L is a valid locked_persona on a persona P. If *verify* is passed successfully, the SP knows two facts. First, the individual D is indeed certified by PP to use P. Second, L is a proof that D has requested a service from SP.

Operations at DAs

$setupAtDA : security\_param \rightarrow DAPukey \times DAPrkey$

setupAtDA generates the public-private key pair for the DA.

$trace : locked\_persona \times DAPrkey \rightarrow persona$

trace is used by a DA to trace a locked_persona back to a persona.

We use the notion of hidden ID-based signatures to construct a system that support personas. The hidden ID-based signature scheme introduced in [12] is an ID-based signature scheme with the following property: signed messages are verifiable without the public key (identity) of the signer. Only the public key of the identity provider is needed. The scheme employs an identification protocol and turns the protocol into a signature scheme using the Fiat-Shamir heuristic [10].

A.1.1 The hidden ID-based signatures scheme

The scheme splits the role of the identity provider into: an identity manager and an opening authority. The identity manager issues certificates to individuals, while the opening authority may open the signatures generated from these certificates. Opening a signature refers to the process of extracting the public key of the signer. The scheme provides these four operations:

`Setup`. Initializes the public/private key pair of both Identity Manager (IDP) and de-anonymizing Authority (DA).
`Registration`. The IDP registers an individual by issuing a certificate, which is a signature on that individual identity produced by the IDP private key.
`Reg. Check`. The individual checks whether the identity and certificate pair are valid with respect to each other.
`Sign`. Signatures are generated as follows:

- The individual commits the identity.
- The individual commits the certificate.
- The individual uses a Σ-protocol to prove the knowledge of the value of the committed identity and certificate, and that the certificate is a signature on that identity.
- The variables of the Σ-protocol are hashed along with the message.
- The committed identity, committed certificate, hash, and message are sent to the verifier.

`Verify`. The verifier uses the IDP public key to check the signature is valid and that the signer certificate and identity are encrypted with the DA public key.
`Open`. The DA uses its private key to decrypt and extract the signer's committed identity from a valid signature.

A.2 Multishow unlinkability

The previous scheme cannot be used as is to implement the requirements of a P-IMS. The signer can prove the possession of a certificate from the IDP (PP in P-IMS), but nothing beyond that. The scheme, therefore, needs modification to accommodate IMS functionalities, e.g., showing attributes. The scheme uses individuals' real identities, which puts these identities in jeopardy if the committed values are maliciously decrypted. We modify the scheme by changing the following operations.

`Setup`. Setup is composed of *setupAtPP* and *setupAtDA*. *setupAtPP* is used to generate the required keys for issuing personas (PP public-private key pairs). *setupAtDA* is used to generates the keys for opening signatures (DA public-private key pairs). Setup also initializes the public parameters.

First, Setup generates $(p, g, \mathbb{G}, \mathbb{G}_2, e)$, where $\mathbb{G}$ is a cyclic group of prime order p, with a generater g, and e is a bilinear map, $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_2$. PP public key is the pair ($X = g^x$, $Y = g^y$), where x and y are random elements in $\mathbb{Z}_p$. PP private key is the pair (x, y). DA public key is given by (u, v, w), where w is a random element in $\mathbb{G}$ and $w = u^b = v^d$, b and d are random elements in $\mathbb{Z}_p$. DA private key is (b, d).

`Registration`. Identifiers which encode an individual attributes are used instead of the real identity. An identifier is composed of the pair (*pseudonym*, *attributes*), where pseudonym is a random number to distinguish between individuals, and attributes are what the individual claims. It is worth mentioning that attributes are represented not as absolute values, but as assertions (e.g., age > 18, instead of age=32). The rationale behind this is to reduce the effects of data-fusion and knowledge-discovery techniques. For simplicity, assume that the i bits of the identifier encodes the pseudonym, while the remaining j bits encodes the attributes.

The PP can issue a certificate to an individual by generating and signing two identifiers (*base_identifier*, *full_identifier*) for that individual. We use the signature scheme presented in [3], which is the same scheme used in [12]. The pseudonym part of *base_identifier* and *full_identifier* are equal. The attributes part of *base_identifier* is assigned to 0, i.e., no attributes, while the attributes part of *full_identifier* is assigned to the encoding of the attributes that the individual is entitled to. The mapping between the identifiers and the real identity of the individual is securely stored at the PP. In Section A.6.1, we will see how to relax this assumption resulting in a more privacy-preserving system. The PP sends the two identifiers and the signature on them to the individual. The identifiers constitute the persona, while the signature (s_{base}, s_{full}) on them is the secret.

$$\begin{gathered} persona = \{base\_identifier,\ full\_identifier\ \} \\ secret = \{s_{base} = g^{(x+y*base\_identifier+r_{base})^{-1}}, \\ s_{full} = g^{(x+y*full\_identifier+r_{full})^{-1}}\} \end{gathered}$$

Note that r_{base} and r_{full} are random elements in $\mathbb{Z}_p$.

`checking Registration`. An individual may check the validity of her (persona, secret) pair by checking if the following hold $e(s_{base}, Xg^{base\_identifier}Y^{r_{base}}) = e(g, g)$ and $e(s_{full}, X g^{full\_identifier}Y^{r_{full}}) = e(g, g)$

`Sign`. The individual uses the *base_identifier* to sign messages that require no attributes. For example, an individual affiliated with a university authenticates to the ACM Digital Library, which requires nothing more than that the individual is affiliated with that university.

If the individual needs to show attributes, both identifiers are needed. Signing a message by a persona requires these steps (the case requiring both identifiers):

- The individual uses the encryption scheme of [4] to commit both identifiers in (U, V, W).

$$U = u^l, \qquad V = v^k$$

$$B = w^{l+k}\, g^{base\_identifier}, \quad F = w^{l+k}\, g^{full\_identifier} \tag{1}$$

  Where l and k are random numbers in $\mathbb{Z}_p$. This will help us draw a comparison between B and F.

- The individual commits the secret (s_{base} and r_{base}) in S_{base}, R_{base}. $S_{base} = g^{r_1} s_{base}$ and $R_{base} = g^{r_2} h^{r_1} Y^{r_{base}}$, where r_1, r_2 are random elements in $\mathbb{Z}_p$, and h is a random element in $\mathbb{G}$. The secret (s_{full}, r_{full}) is also committed in S_{full}, R_{full}.

- The individual uses an extended version of the Σ-protocol (described below) to prove the knowledge of the committed value of the persona, secret and that the secret is a valid signature on the identifiers (persona).

- The variables of the extended Σ-protocol are hashed along with the message.

- The committed persona, committed secret, hash, and the message are sent to the SP.

- The attributes are also sent to the SP.

The extended Σ-protocol
The original protocol of [12] proves the knowledge of the signer's identity and certificate. We use the same protocol to prove the knowledge of the individual's persona and secret. The difference is that added variables are needed for the persona and the secret, instead of one identity and a certificate. Readers are refered to [12] for the protocol details.

`Verify`. The first step in the verification is running the verify operation of [12].

The second step of the verification process operates on the committed values of the identifiers (persona). The SP encodes the attributes sent by the individual as $g^{attributes}$. The pseudonym bits are the same in both identifiers; and the only difference is in the attribute bits. Therefore, $attributes = full\_identifier - base\_identifier$. The SP needs to check whether this equality holds:

$$B\, g^{attributes} = F \tag{2}$$

If it does, then the individual is certified by the PP to have the attributes.

We need to prevent an individual from swapping B and F, which would enable her to claim $-attributes$, instead of $attributes$. This is done by appending two bits to both identifiers at the most significant part. That is, *base_identifier* bits become the binary string

of 00 appended to *base_identifier* bits. *full_identifier* bits become the binary string of 01 appended to *base_identifier* bits. Equation 2 becomes:

$$B\, g^{second\_bit\_set}\, g^{attributes} = F \tag{3}$$

second_bit_set is a binary string with the same number of bits as the *base_identifier*. All bits are assigned to 0, except for the 2nd most significant bit, which is assigned to 1 (i.e., 010...0).

`Open`. The private key of the DA is used to extract $g^{base\_identifier}$ and $g^{full\_identifier}$ from the commitments. The signature opening algorithm of [12] is used to extract *base_identifier* and *full_identifier* which are sent to the PP. Finally, the PP maps the identifiers back to the real identity.

A.3 Selective release of attributes

Instead of having two identifiers: *base_identifier* and *full_identifier*, the PP can provide an individual with many identifiers. Selective release of attributes is achieved by providing an individual with an identifier per attribute or group of attributes. Let *age_attribute_identifier* be of the same bit-length as *full_identifier*, and the *pseudonym* bits be equal. However, all the *attribute* bits are 0s except for the bits encoding the age. The bit string 01 is appended to *age_attribute_identifier* as the case with *full_identifier*. An individual with *age_attribute_identifier* certified by PP can use the sign and verify operations to show the age only as follows. Let FA be as in Equation 4.

$$FA = w^{l+k}\, g^{age\_attribute\_identifier} \tag{4}$$

If Equation 5 holds, then the individual is certified by the PP to have *age_attribute*.

$$B\, g^{second\_bit\_set}\, g^{age\_attribute} = FA \tag{5}$$

There are two ways to show multiple attributes. The sign and verify operations can be repeated with different identifiers, or the sign and verify functions can be called once, but with extra variables to accommodate all the committed values of identifiers.

A.4 Relationship verification

An identifier is composed of a pseudonym part and an attribute part. Having different identifiers with equal pseudonyms means that the identifiers belong to the same individual. Just as increasing the number of identifiers achieves selective release of attributes, increasing the number of parts of each identifier achieves relationship verification. Let's add a third

part, called a *relation*. An identifier becomes the composition of a pseudonym part, a relation part, and an attribute part.

$$identifier\ bits = (pseudonym\ bits)(relationship\ bits)(attributes\ bits)$$

Individuals may prove the existence of relation between identifiers.

Let *base_identifier1* and *base_identifier2* be two *base_identifiers* issued by a PP for individuals D1 and D2, respectively. Let the *pseudonym* bits of both identifiers be equal, and there be a relation between D1 and D2, for example, D1 is the boss of D2. PP encodes that relationship by giving D1 and D2 different values for the *relationship* bits. For example, relation bits of D1 is set to x and the one for D2 is set to y (this encoding is just a trivial one used for simplicity of discussion). Note that the *attribute* bits are assigned to 0 in both identifiers.

An SP can check if D1 is the boss of D2 by asking D1 and D2 to sign a message with their *base_identifiers* and verifying Equation 6. *B_D1* and *B_D2* are computed as B is computed for the case of one individual. $relation = y - x$, and its appended-to bits of 0s of length a (the attribute bit size).

$$B\_D1\ g^{relation} = B\_D2 \tag{6}$$

In the same manner, we can specify relations that involve several individuals. In other words, personas can model graphs, where the nodes are individuals and the edges are their relations. We achieve this by computing an adjacency matrix for the required graph. Each cell encodes the relation between two individuals: the individual corresponding to the column of the cell, and the individual corresponding to the row. Thus, each row encodes the relationship between an individual and the remaining individuals. Each row can be treated as a relation given by $relation = y - x$. The only difference is that $relation$, y, and x are vectors composed of n components rather than scalars, where n is the number of individuals involved.

Now, we explain in details how a IP provides a set of individuals D with a set of personas S allowing them to prove a set of relations R. Let D_i denotes the i^{th} individual, R_i denotes the set of relations of the i^{th} individual, and S_i denotes the i^{th} persona. Clearly, D and R forms a graph G. The individuals in D corresponds to the nodes of G, whereas the relations in R corresponds to the edges of G. Let A be the adjacency matrix of G, where A_i denotes the i^{th} row, A_i^j denotes the cell at row i and column j. A_i is set to R_i.

Algorithm 1 takes A as input and produces S as output. ¿From A we compute $\acute{A}$, where the *relation* part of S_i identifier is set to $\acute{A}_i$. Then, all S_i are generated to get S.

Note that the *persona* part is 0 for all S, and that they have the same value for *pseudonym*. The $\|$ symbol refers to concatenation. The algorithm above encodes the relationships of D_i, $1 \leq i \leq n-1$. The relationships of D_n can be inferred from other relationships (undirected graphs). For directed graphs, a new persona S_{n+1} is needed to compensate. S_{n+1} is computed

as other S_i in the algorithm. The IP finally sends S to D, where each D_i receives an S_i. In case of directed graphs, D_n receives two personas (S_n and S_{n+1}).

Now we turn to how the individuals prove R to SP using S. They sign the same message using their personas and send the resulted locked personas to the SP. Let LS denotes the set of the locked personas. They also send R to the SP, where the SP encodes R as A. The SP then runs Algorithm 2. It is clear from $\acute{A}_i^j = A_{i-1}^j + \acute{A}_{i-1}^j$, that $A_i^j = \acute{A}_{i+1}^j$ - $\acute{A}_i^j$ holds. That is, the relationships of the i^{th} individual can be recovered from the $\acute{A}_i$ and $\acute{A}_{i+1}$. The algorithm uses LS_{i+1}, LS_i, and A_i as input to *VerifyRelation*. If any instance of *VerifyRelation* does not pass, the algorithm outputs reject. Otherwise, it outputs accept.

Input: A
Output: S
$\acute{A}_1 = 0$
foreach $\acute{A}_i$ *in* $\acute{A}$, $i \neq 1$ **do**
  $temp = \sqrt{(A_i^1)^2 + (A_i^2)^2 + ... + (A_i^n)^2}$
  $\acute{A}_i = temp + \acute{A}_{i-1}$
end
$pseudonym = random$
$attribute = 0$
foreach S_i *in* S **do**
  $relation = \acute{A}_i$
  $identifier = pseudonym \parallel relation \parallel attribute$
  $secret$ = IP signature on $identifier$
  $S_i = \{ identifier, secret \}$
end

Algorithm 1: Generating personas based on an adjacency matrix

Input: A, LS
Output: *accept*, *reject*
foreach LS_i *in* LS **do**
  $relation = \sqrt{(A_i^1)^2 + (A_i^2)^2 + ... + (A_i^n)^2}$
  if *VerifyRelation*$(LS_{i+1}, LS_i, relation) = reject$ **then**
    output reject
    hault
  end
end
output accept

Algorithm 2: Verifying locked personas against an adjacency matrix

Table 1: Mapping cryptographic constructs to the P-IMS operations

P-IMS		The Construction
`PP`	→	`IDP`
`DA`	→	`DA`
`Persona`	→	Identifiers
`Secret`	→	Signed Identifiers
`Attributes`	→	Attributes part of an identifer
`Locked_Persona`	→	Signature
`Wrap`	→	`Registration`
`Show`	→	`Sign`
`SelectiveShow`	→	`Sign` as in Section A.3
`Verify`	→	`Verify`
`VerifyRelation`	→	`Verify` as in Section A.4
`Trace`	→	`Open`

A.5 Accountability of users

One disadvantage of Hidden ID-based Signatures is the time that the identity provider needs to open a signature to reveal the identity of the signer. The time is super-polynomial with regard to the identity length. In our case, however, this turns out to be an excellent feature. We want the DAs to be discouraged from building profiles of users.

A.6 Meeting P-IMS requirements

The building blocks of the P-IMS is now ready and given by Table A.6. The table shows each operation/concept in P-IMS and its equivalent construction that uses and extends the hidden ID-based signatures. `Show` and `Verify` allows for anonymity and multishow unlinkability of attributes. Selective release of attributes is achieved by `SelectiveShow`, whereas relationship verification is achieved by `VerifyRelation`. `Trace` implements persona traceability.

A.6.1 Nesting personas

As we noted in [11], individuals may nest personas. Nesting personas allows individuals to hide their identities behind layers of personas. All PPs who generated a nested persona, except for the first one, do not have the ability to find the individual behind that persona. To find the individual, each PP must execute *trace* at the previous PP. It is worth mentioning that a DA does not execute trace unless the PP or SP provides it with a proof of a misuse.

Technically, nesting personas moves attributes from one persona into another. An individual, D, may use *wrap* at a PP, say PP1, to receive a persona. D then executes *show* with that persona, and sends the locked persona to PP2, another PP. If PP2 trusts PP1, *wrap* is executed at PP2 to generate a new persona for D. The new persona may contain a subset of the attributes specified in the old persona (depending on how many attributes D has disclosed). Note that PP2 does not check against D and does not know anything about her.

An alternative approach is to not move the attributes, but to store the locked persona of the old persona in the new persona. This way, SPs can verify the whole chain of PPs in a persona and there will be no need to trust the last PP in a chain as in the previous case. This alternative, however, contradicts with multishow unlinkability requirement. The reason for that is the locked persona of a persona, once stored, becomes like a tag. Repetitive usage of the new persona will be linkable. Therefore, we avoid this alternative and stick with the original approach.

A.6.2 Implementation and testing

We have implemented and tested the system with the help of the Pairing-based Cryptography (PBC) library [14]. The PBC is a free library written in C and it provides the necessary functions to write programs that handle pairings. Elliptic curve generation, elliptic curve arithmetic, and pairing computation are provided as routines for programmers. Different types of pairings are available. The library may perform a pairing computation in an average time of 11 ms on a 1 GHz Pentium 3 machine.

A.6.3 Alternative solution

An alternative solution would be extending the notion of ID-based group signatures as in [5]. Group signatures also allow for verifying signatures without the signer's public key.

A.7 Correctness and Security

The correctness and security of P-IMS is drawn from the correctness and security of the underlying hidden ID-based signatures. A P-IMS is correct if the following three conditions hold. The checkWrap operation always succeed when executed on a valid (persona, secret) pair.

$$
\begin{aligned}
&\mathit{Probability}[\\
&\quad (\mathit{persona}, \mathit{secret}) \leftarrow \mathit{wrap}(\mathit{attributes}, \mathit{proof}, \mathit{PPPrkey})\\
&\quad \mathit{true} \leftarrow \mathit{check\_wrap}(\mathit{persona}, \mathit{secret}, \mathit{PPPukey})] = 1
\end{aligned}
$$

The verify operation always succeed when executed on a valid proof.

$$
\begin{aligned}
&\mathit{Probability}[\\
&\quad (\mathit{persona}, \mathit{secret}) \leftarrow \mathit{wrap}(\mathit{attributes}, \mathit{proof}, \mathit{PPPrkey})\\
&\quad \mathit{true} \leftarrow \mathit{check\_wrap}(\mathit{persona}, \mathit{secret}, \mathit{PPPukey})\\
&\quad \mathit{proof} \leftarrow \mathit{show}(\mathit{persona}, \mathit{secret}, \mathit{PPPukey}, \mathit{DAPukey})\\
&\quad \mathit{true} \leftarrow \mathit{verify}(\mathit{proof}, \mathit{PPPukey}, \mathit{DAPukey})] = 1
\end{aligned}
$$

The trace operation always extracts the persona used by the show operation to generate a proof verifiable by the verify operation.

$$
\begin{aligned}
&\mathit{Probability}[\\
&\quad (\mathit{persona}, \mathit{secret}) \leftarrow \mathit{wrap}(\mathit{attributes}, \mathit{proof}, \mathit{PPPrkey})\\
&\quad \mathit{true} \leftarrow \mathit{check\_wrap}(\mathit{persona}, \mathit{secret}, \mathit{PPPukey})\\
&\quad \mathit{proof} \leftarrow \mathit{show}(\mathit{persona}, \mathit{secret}, \mathit{PPPukey}, \mathit{DAPukey})\\
&\quad \mathit{true} \leftarrow \mathit{verify}(\mathit{proof}, \mathit{PPPukey}, \mathit{DAPukey})\\
&\quad \mathit{persona} \leftarrow \mathit{trace}(\mathit{proof}, \mathit{DAPukey}, \mathit{DAPrkey})] = 1
\end{aligned}
$$

A P-IMS is secure against misidentification attacks, if the probability of an adversary succeeding in the following game is negligible. In this game, the adversary has access to wrap_oracle, which executes a wrap operation and returns the resultant (persona, secret) pair. The adversary has access to show_oracle, which executes a show operation and returns the resultant proof. The adversary wins the game if it produces a valid proof that is either not traceable to a persona, or is traceable but the adversary has not used wrap_oracle to receive that persona and she did not use show_oracle to produce that proof.

wrap_oracle(*attributes*)
  (*persona*, *secret*) ← *wrap*(*attributes*, *proof*, *PPPrkey*)
  Personas ← {*persona*} ∪ *Personas*
  return (*persona*, *secret*)
show_oracle(*persona*)
  proof ← *show*(*persona*, *secret*, *PPPukey*, *DAPukey*)
  Proofs ← {*proof*} ∪ *Proofs*
  return *proof*
misidentification_game()
  proof ← *Adversary*(*wrap_oracle*, *show_oracle*)
  if(*true* ← *verify*(*proof*, *PPPukey*, *DAPukey*) **AND**
  Φ ← *trace*(*proof*, *DAPukey*, *DAPrkey*))
    Adversary *wins*
  elseif(*true* ← *verify*(*proof*, *PPPukey*, *DAPukey*) **AND**
  persona ← *trace*(*proof*, *DAPukey*, *DAPrkey*) **AND**
  persona ∉ *Personas* **AND** *proof* ∉ *Proofs*)
    Adversary *wins*
  else *Adversary* *loses*

A P-IMS is secure against adaptive chosen-cyphertext attacks (CCA2), if the probability of an adversary succeeding in following two game is $0.5 + \epsilon$, where ϵ is negligible. The adversary has access to trace_oracle, which reveals the persona used to generate a proof. The adversary is presented with a proof and two personas, in which one of personas was used to generate the proof. The adversary wins the game if it guesses the right persona. Of course, the adversary is constrained from using trace_oracle on the presented proof. trace_oracle$_\alpha$ refers to that constraint.

trace_oracle($proof$)
  $persona \leftarrow trace(proof, DAPukey, DAPrkey)$
  $return \quad persona$
CCA2_game()
  $(persona_1, secret_1) \leftarrow wrap(attributes_1, proof_1, PPPrkey)$
  $(persona_2, secret_2) \leftarrow wrap(attributes_2, proof_2, PPPrkey)$
  $r \leftarrow random \quad from \quad \{1, 2\}$
  $proof \leftarrow show(persona_r, secret_r, PPPukey, DAPukey)$
  $challenge \leftarrow \{proof, persona_1, persona_2\}$
  $guess \leftarrow Adversary(trace\_oracle_\alpha, challenge)$
  if$(guess = persona_r)$
    $Adversary \quad wins$
  else $Adversary \quad loses$

The proofs of correctness and security of the hidden ID-based signature scheme are presented in [12]. The scheme is secure against misidentification and CCA2 attacks under the Strong Deffie-Helman (SDH) and Decisional Linear Deffie-Helman (DLDH) assumptions in the random oracle model.

The wrap operation of P-IMS is a composition of two register operations in the signature scheme. The show operation of P-IMS is a composition of two sign operations in the signature scheme. The remaining operations of P-IMS are the same in the signature scheme. Let an adversary A has the ability to launch successful misidentification and or CCA2 attacks on P-IMS. A can be used as an adversary to launch successful misidentification and or CCA2 attacks on the signature scheme. Therefore, the P-IMS is correct and secure against misidentification and CCA2 attacks under the SDH and DLDH assumptions in the random oracle model.

B Sample Scenario

In this section we illustrate the use of presented system through a sample scenario. Let Tom and Jennifer be a couple applying for a joint back account at Bank_B. The bank and the government run the presented system and the government has issued two personas for Tom and Jennifer. The following scenario shows how Tom and Jennifer use their personas to subscribe to a joint account. Please refer to Figure 8.

Recall that to specify a relation, Tom and Jennifer must have the same pseudonym part

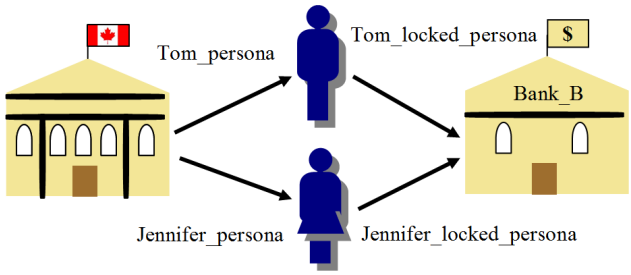


Figure 8: The illustration of the scenario

and they need the relation part present in the identifiers (Section A.4). We set the relation part of Tom to 0 and the one of Jennifer to 1 (The type of the relation, which is marriage, should be encoded too, but we omit this detail for clarity). The couple's personas are:

$$
\begin{aligned}
Tom\_persona &= \{Tom\_base\_identifier = pseudonym, 0, 0...0, \\
&\quad Tom\_full\_identifier = pseudonym, 0, Tom\_attribute\} \\
Tom\_secret &= \{Government\ signature\ on(Tom\_base\_identifier), \\
&\quad Government\ signature\ on(Tom\_full\_identifier)\} \\
Jennifer\_persona &= \{Jennifer\_base\_identifier = pseudonym, 1, 0...0, \\
&\quad Jennifer\_full\_identifier = pseudonym, 1, Jennifer\_attribute\} \\
Jennifer\_secret &= \{Government\ signature\ on(Jennifer\_base\_identifier), \\
&\quad Government\ signature\ on(Jennifer\_full\_identifier)\}
\end{aligned}
$$

The couple use `Show` operation of the presented system, at their machine, to generate the required locked personas. The couple send the locked personas to authenticate to the bank as a couple.

$$
\begin{aligned}
Tom\_locked\_persona &= Show(Tom\_persona, Tom\_secret) \\
Jennifer\_locked\_persona &= Show(Jennifer\_persona, Jennifer\_secret)
\end{aligned}
$$

Bank_B uses `VerifyRelation` operation to verify that the couple have two personas from the government and that they are a couple. couple_relation is the encoding of the couple relation as agreed on between the government and SPs.

$$
VerifyRelation(Tom\_locked\_persona, Jennifer\_locked\_persona, couple\_relation)
$$

The couple may reveal more attributes to gain extra features. They do so using the selective release of attributes. If the Bank_B is satisfied by the attributes, it generates one or two personas for the couple (depending on the business logic). The couple may use these personas at all SPs that trust Bank_B.

$$Tom\_persona\_Bank_B = Wrap(Tom\_locked\_persona)$$
$$Jennifer\_persona\_Bank_B = Wrap(Jennifer\_locked\_persona)$$

For the sake of argument, assume that Bank_B feels it has been cheated by the couple or the couple are refusing to pay their debt. Accountability is enforced by Bank_B sending Tom_locked_persona and Jennifer_locked_persona to the DA (or to the government if the government and DA are the same entity). The DA uses `Trace` operation to extract the personas from the locked_personas. The DA then sends the Tom_persona and Jennifer_persona to the government for legal action.

$$Tom = Trace(Tom\_persona\_Bank_B)$$
$$Jennifer = Trace(Jennifer\_persona\_Bank_B)$$

References

[1] Identity, privacy and the need of others to know who you are: A discussion paper on identity issues. Technical report, Office of the Privacy Commissioner of Canada, September 2007. Retrieved from `http://www.privcom.gc.ca/information/pub/id_paper_e.pdf`, on March 2009.

[2] Byron Acohido. Hackers breach heartland payment credit card system, January 2009. USA Today, retrieved from `http://www.usatoday.com/money/perfi/credit/2009-01-20-heartland-credit-card-security-breach_N.htm`, on April 2009.

[3] Dan Boneh and Xavier Boyen. Short signatures without random oracles. In *Proceedings of the 24th International Conference on the Theory and Applications of Cryptographic Techniques*, pages 56–73. Springer-Verlag, 2004.

[4] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In *Proceedings of the 24th International Conference on the Theory and Applications of Cryptographic Techniques*, pages 41–55. Springer-Verlag, 2004.

[5] Xavier Boyen and Brent Waters. Compact group signatures without random oracles. In *Proceedings of the 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 427–444. Springer-Verlag, 2006.

[6] Jan Camenisch and Els Van Herreweghen. Design and implementation of the idemix anonymous credential system. In *Proceedings of the ACM Conference on Computer and Communications Security*, pages 21–30. ACM Press, 2002.

[7] Jan Camenisch, Abhi Shelat, Dieter Sommer, Simone Fischer-Hübner, Marit Hansen, Henry Krasemann, Gérard Lacoste, Ronald Leenes, and Jimmy Tseng. Privacy and identity management for everyone. In *Proceedings of the Workshop on Digital Identity Management*, pages 20–27. ACM, 2005.

[8] D. Chappell, Introducing Windows CardSpace, retrieved from `www.msdn.microsoft.com/en-us/library/aa480189.aspx`, on May 2008.

[9] U-Prove SDK overview. white paper, Credentica Inc, 2007. retrieved from `www.credentica.com/files/U-ProveSDKWhitepaper.pdf`, on May 2008.

[10] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Proceedings of the Conference on Advances in Cryptology*, pages 186–194. Springer-Verlag, 1986.

[11] Mohammed Hussain and David B. Skillicorn. Persona-based identity management: A novel approach to privacy protection. In *Proceedings of the 13th Nordic Workshop on Secure IT Systems*, pages 201–212. Technical University of Denmark, 2008.

[12] Aggelos Kiayias and Hong-Sheng Zhou. Hidden identity-based signatures. In *Proceedings of the 11th International Conference on Financial Cryptography and Data Security*, pages 134–147. Springer-Verlag, 2008.

[13] Liberty alliance project specifications. retrieved from `www.projectliberty.org/liberty/specifications__1`, on May 2008.

[14] Ben Lynn. Pairing-based Cryptography Library, retrieved from `http://crypto.stanford.edu/pbc/`, on November 2008.

[15] Teruko Miyata, Yuzo Koga, Paul Madsen, Shin-Ichi Adachi, Yoshitsugu Tsuchiya, Yasuhisa Sakamoto, and Kenji Takahashi. A survey on identity management protocols and standards. *Transactions on Information and Systems*, E89-D(1):112–123, 2006.

[16] Shibboleth. retrieved from `www.shibboleth.internet2.edu/`, on May 2008.

[17] Bob Sullivan. 40 million credit cards exposed, June 2005. MSNBC, retrieved from `http://www.msnbc.msn.com/id/8260050/`, on April 2009.

[18] Web services federation language. retrieved from `www.ibm.com/developerworks/library/specification/ws-fed/`, on May 2008.