Computer Networks 56 (2012) 3834–3848

Contents lists available at SciVerse ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Information privacy?!

Jan Camenisch

IBM Research – Zurich, Säumerstrasse 4, CH-8803 Rüschlikon, Switzerland

A R T I C L E   I N F O

Article history:
Available online 7 November 2012

Keywords:
Privacy
Anonymous credentials
Security
Cryptography

A B S T R A C T

When using the Internet, we reveal much personal information both willingly and inadvertedly. Companies use this information for targeted advertisement and thereby to finance the services they offer to users. The mechanisms used today to protect users' personal information are lacking resulting in far too frequent privacy and security breaches that put the users at risk. In this article we argue that applications on the Internet should be built with privacy and security as a mandatory requirement, then provide an overview of the state of the art in privacy-enhancing mechanisms, and conclude with a roadmap towards a privacy-enhanced digital world, and pointing out a number of challenges that need to be solved.

1. Introduction

The Internet has drastically changed in the last decade. It used to be a network connecting mainly universities allowing researchers to exchange scientific information and people were thrilled creating their own home-pages describing and making available their research results. It was hard to navigate in this space and so, many people maintained collections of links to useful information (such as home-pages of other people working in the same field) and made these collections available to other people. Search engines such as Altavista and later Google revolutionized the use of the Internet. It was no longer painful to look for information and necessary to maintain hundreds of links, one could just "google" for it. This has made the Internet accessible to everyone and attractive for selling goods over the Internet. One could search for product reviews on-line and increasingly buy products on-line as well. Apart from making the Internet easy to use, Google has also changed the way of financing services offered to users: free through targeted ads. New websites and services were built to attract people and to earn money by showing them advertisements. The better an advertisement can be targeted the more valuable it is – in other words, the more data about a user is available, the more money can be made with presenting an ad to the user. Indeed, personal information has become the "new currency" on the Internet.

Having become accessible and usable to a wide audience, the information exchanged via the Internet has also drastically changed: from exchanging scientific and professional information to enormous amounts of personal information. Blogging and twittering has become very popular and social network sites such as Facebook and Foursquare are for many the central place to connect and converse with their friends. Thereby, people reveal lots of personal information. It is not uncommon to learn things about friends from their profiles on some social network that one had not known and that they would probably never have told one.

Finally, the widespread adoption of powerful touchscreen mobile phones that allow users to install small applications (apps) has further revolutionized the use of the Internet and also further increased the amount and quality of personal information exchanged over it. Apps have changed the way services are built and offered. Services are no longer monolithic websites by a single provider but are rather built from high-level building blocks and often aggregate different services into a single one. Whereas the IP protocol and HTML used to be the basis upon which services were built, now apps and websites can make use of a whole infrastructure of services. Examples of such infrastructure components include the

E-mail address: jca@zurich.ibm.com

1389-1286/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.comnet.2012.10.012

iOS and the android APIs (e.g., in app payment or access location data) and APIs to access user data on Facebook or Foursquare. These components form a new higher level basis from which better and powerful services can be built quickly and easily.

The Internet was meant to be an open environment and with little security, let alone privacy protection, in mind. While this openness was essential for the Internet to grow, it comes with risks. Indeed, the absence of adequate protection has become very apparent: every day major newspapers report privacy breaches. The majority of the newspaper articles report on company X having lost or being stolen millions of users' data such as passwords or credit card information. An interesting and very different case were the reports on the "Girls Around Me" app (e.g., [6]). The app showed one pictures and profile information about women near one's location. Here, no data was lost or stolen but rather the app aggregated publicly available information, in this case the location information from Foursquare and the profile information from Facebook. The main reaction to this app was that it was very dangerous because it provided a tool for stalkers and rapers. Few people realized that this was a case of "shooting the messenger" and that it should rather be considered a "wake up call for privacy." This app well demonstrates what is possible with all the (personal) information available via the Internet and what can be done with it.

These reports only hint at the risks that come with using the Internet and its services and what criminals could potentially do with all the data. It also shows that the security mechanisms currently employed on the Internet are not sufficient. In fact except from TLS/SSL, the security mechanisms used are "patch on" solutions such as virus filters and firewalls.

As our personal data gets dispersed and mined and the security mechanisms deployed lacking or wrongly applied, can we hope to achieve privacy? It has been argued that privacy in the digital age is unattainable and statements such as McNealy's "*You already have zero privacy-get over it*" are common. Indeed, achieving privacy seems nearly impossible. For instance, building an over-lay network on top of the Internet that provides true anonymous communication is very hard: it has been shown that the physical fingerprint of a device can remotely be recognized even when communicating with it trough a number of routing hops [50]. These attacks however assume a very powerful and dedicated adversary and probably only institutions such as NSA are able mount this kinds of attacks. Indeed, we cannot realistically hope to achieve perfect privacy protection on the Internet, just as protection against professional burglars in the physical world is out of reach for normal people. However, protection that is good enough to prevent almost all mischief is definitely possible and future infrastructures and services could be built with security and privacy as initial design criteria. Thereby, three principles should be followed.

1. An application should be designed so that only the minimal amount of (personal) information gets revealed to each party that is necessary for the party to perform its task.
2. Users need to be able to understand and control the usage of the information they have released.
3. All information related to users must be encrypted, both at rest and in transit.

At first glance these principles often seem hard to achieve or even to contradict functional requirements of applications. For instance, how can one do access control without identifying the requestor? Or, how can one counter denial of service attacks when communication is anonymous? Modern cryptography, fortunately, provides answers to such questions and in this article we therefore study different technologies and how they can be used to make the Internet a safer and privacy–friendly place.

2. Data minimization and security at the same time

More than thirty years ago, cryptographer David Chaum realized that a digital society will require mechanisms to protect the privacy of users. Addressing this need, he suggested many cryptographic techniques such as for instance privacy-protecting communication protocols [32] or anonymous e-cash schemes [33]. David's seminal work has inspired many other researchers and a very active research community has grown. Innumerable cryptographic mechanisms that can be used to provide and protect digital privacy have been invented; here we can only give a few examples on how which mechanisms can be used. To this end, we divide the mechanisms into three categories. The first type of mechanism is concerned with providing privacy at the network layer, to ensure that communication channels can be established without revealing identifying information such as IP addresses. Once such communication has been established, the second type of mechanism comes into play. They allow users to reveal only information that is necessary for the task at hand. The third category are mechanisms that implement special purpose applications. Again, there are far too many of those and thus we give a couple of examples to show what can be done.

3. Data minimization at the network layer

Most communication over the Internet neglects security and privacy protection: messages are sent in the clear and the identities of the receiver and the sender are not protected either. Thus, this information is leaked to all parties routing a message and can even be altered by them!

The mechanisms for the protection of the contents are rather well understood, are implemented in many applications such as web browsers (e.g., TLS) and email clients (e.g., S/MIME), and are increasingly used. In contrast, only a few means are available to protect the identities of the sender and receiver of messages and these are hardly used in practice. We here give a brief overview of the state of the art of the main approaches in this area (the interested reader is referred to the literature for extensive overviews [42,60]).

The main idea of protecting the sender and receiver of messages is relatively simple [32]. When a sender Alice sends a message to a receiver Bob via a message router Rob, she first encrypts the message under Bob's public key and then encrypts the results together with Bob's name under Rob's public key. She then sends this final encryption to Rob who decrypts it, sees another encryption plus Bob's name and thus forwards this encryption to Bob. Upon receiving it, Bob decrypts and then can read the message from Alice. Thus, from the communication packets sent between Alice, Rob, and Bob one can only tell that Alice sent some message to Rob and that Rob sent some message to Bob. Of course, if these two where the only messages sent to and from Rob and it is known that Rob is just a routing party, it becomes obvious that Alice has sent a message to Bob (but the content of the message will still be secret). Even if multiple users route their messages via Rob and Rob forwards them in the order received, the sender and recipient of messages are revealed by observing the network traffic. Therefore, the order of how messages are received and sent must be different, i.e., messages should be cached and mixed before they are sent on by Rob. Because of this, Rob is often referred to as a mix-server and the network as a mix/network.

This approach works fine if Rob is fully trusted to (1) keep secret how he mixed the messages and (2) send on all received messages, i.e., not to drop or replace messages. To lessen the trust to be put into Rob w.r.t. the first issue, one can just use multiple routers in a row, hoping that at least some of them keep the input–output relation secret. Addressing issue (2) is more involved and different solutions are preferable for different use cases. One way is to require the routers to cryptographically prove that their messages sent are a permutation of the messages received, which might be very costly, however. If such proofs are too costly as is the case for real-time communication, one is left with some form of resending the message, possible using different routers. In the following we briefly discuss the state of the art for both cases together with typical use scenarios.

3.1. Provable mix-networks

A typical scenario where one wants to use provable mix-networks is anonymous voting. Here, a voter encrypts her message encoding her vote and some cryptographic information relating to the voting scheme under the last mix-server's public key, the results then under the second last mix-server's public key, etc. The first server then waits until all voters have sent their votes, then removes the first layer of encryption, shuffles the resulting decryption, and then publishes all the messages received, the shuffled decryptions, together with a cryptographic proof that the shuffled decryptions are correct decryptions of all the messages received. The next mix-server proceeds analogously with the shuffled decryptions of the first mix-server as input messages. This is repeated until the final mix-server and thus all layers of encryptions are removed and the votes of all the voters revealed. Verifying the cryptographic proofs of the mix-servers, everyone can convince themselves that each mix-server correctly forwarded the messages, i.e., no votes were dropped or inserted.

Currently, the most efficient scheme to do the cryptographic proofs is by Groth [47] while Wikström and Groth provide a scheme secure against adaptive attacks [69].

This approach of anonymizing communication provides very good privacy, i.e., if at least one of the mix-server is honest, an adversary observing the network and controlling all dishonest servers cannot tell who sent which message (or cast which vote in this example). More precisely, the adversary cannot tell which of the honest senders sent which message – some of the senders could also be under the adversary's control.

3.2. Mix-cascades and onion routing

Unfortunately, despite offering very good security and properties, provable mix-networks are not suitable for real-time communication because creating the proofs that the messages forwarded are a permutation of the decryption of the messages received is far too inefficient, at least for the known schemes. Luckily in many real-time communication scenarios such as browsing the Internet, one can do with much weaker security properties, in particular, without the guarantee that all messages get routed correctly.

If one just drops the requirement that mix-servers publish the input and output messages and the zero-knowledge proofs one ends up with a mix-cascade network. Apart from the fact that there is no longer a proof all messages are routed, the anonymity properties further change because of the real-time use. In a provable mix-network, all senders send only a single message and the first server waits until all messages have arrived before he begins to route any message. In a real-time scenario this is not feasible: there are just too many potential senders and waiting for all messages would incur prohibitive delays. Thus, the first server waits for sufficiently many messages and then routes messages in batches. This of course significantly reduces the anonymity provided – assuming that an adversary can observe the network and control some of the senders and mix-servers. On the one hand anonymity is reduced just because there are fewer messages to permute. On the other hand an adversary can try to isolate a message from a single user by having dishonest senders send a message at the same time so that all but one message in a batch stem from an adversarial sender. Also, different senders probably send a different number of messages to different receivers and hence just by observing how many messages each sender sends and how many messages each receiver obtains (e.g., because they might send a text, a picture, or a whole movie), it is possible to link some senders and receivers. Thus it is questionable whether in practice batch mixing offers more anonymity than if no mixing is done. On top of this, batch-mixing does not adapt very well to varying traffic loads. If the traffic is too low, the servers have to wait longer for sufficiently many messages before they can process them – otherwise the anonymity offered shrinks.

Dropping both the proofs and the mixing, one ends up with a solution such as the one implemented by JAP [11]. This anonymizer is open for everyone to use (anon.inf.tu-dresden.de/index_en.html). JAP's servers are operated by a number of independent institutions and a number of different cascades are available. The drawback of a fixed cascade as implemented by JAP is that if traffic grows too much, the mix servers will become a serious bottleneck. So-called onion routing overcomes this. Here, there is no-longer a fixed cascade of mix servers but rather there is a list of routers and the sender decides herself which router she wants to use and then encrypts her message accordingly. Also, messages are routed directly when they arrive, i.e., they are not mixed. This is actually what is implemented in TOR (www.torproject.org) [44], the probably best known onion-routing scheme.

There is a large body of research papers on analyzing how much privacy (e.g., what is the probability that a message received can be attributed to the original sender by an adversary) can be achieved by different designs of onion-routing schemes. For instance, what happens if routers do some mixing? Should messages be randomly delayed before they are sent on? Does it help to introduce dummy traffic? The answer to these questions of course depends on the assumptions one makes about the power of a potential adversary. In practice, however, it is probably safe to assume that an adversary is not that powerful (e.g., if not a national security agency) and an approach such as the one by TOR might give very reasonable privacy.

A number of papers have been published that investigate the vulnerability of TOR and related approaches against traffic analysis, i.e., passive observers of the network (see, e.g., [42]). New designs for anonymous routing are still being investigated such as for instance peer-to-peer onion routing [53], where the various senders become routers themselves, or anonymous ad hoc mesh-networks [51]. Indeed, in the last 10 years, the field of anonymous communication has grown considerably and established itself as a research field in its own right and we can expect to see much more research and practical work on anonymous communication in the future.

3.3. Practical aspects

The importance of anonymous communication is getting better recognized, in particular in light of the *Arab Spring*. For instance, it was proposed to use mesh-networks for unobservable communication and first implementations are available (http://commotionwireless.net/). A further indication is that the TOR project is able to raise sufficient funds to pay a few employees and to maintain and progress TOR. Then again, websites such as wikipedia do not like anonymous communication because many fraudulent edits were made by people who used TOR to hide and so wikipedia does currently not allow contributions that come through IP addresses that are known TOR routers (with some exceptions). Such concerns can be addressed however, by adding suitable accountability measures, e.g., by using privacy–friendly authentications, which we discuss in the next section.

4. Data minimization at the authentication and identities layer

Only relatively few and basic tasks can be done completely anonymously. Most often some form of authentication is needed, which requires some (personal) information be revealed. Without loss of generality, all this information consist of attributes of a user. Examples of such attributes include credit card number, age, user name and password, professional qualification, or street address. These attributes need to be validated and certified by a third party such as one's bank, employer, local government, or an educational institution. Today this authentication typically involves X.509 certificates, single sign-on, OpenID, or Facebook Connect. All these means unfortunately result in the user revealing more information than necessary: either the party who wants to authenticate the user learns all the information in a certificate or the party certifying the information learns the transaction details of the user. In the following we summarize the concepts of information-parsimonious authentication and their technical realizations [21].

4.1. Basic concepts

Minimizing the information revealed in their different transactions, requires that the users manage their different identities. In the non-digital world, we have learned to do this: we do not tell everyone the same information and are somewhat careful whom we tell what. In the digital world, this is trickier because digital information gets stored much more persistently than what a human brain can, spreads much easier and accurately than rumors, and can be processed amazingly fast.

To proceed we first need to define what we mean by an identity. One finds very different definitions in the literature. We refrain from a discussion or an overview of the different possible definitions. For our purposes, we view a user's identity as a set of attributes, where we consider any information a party knows about a user to be an attribute of the user. Thus, an identity only exists in connection with a party. Different parties know different things about the same user. Every user has therefore many different identities, possibly even multiple identities with the same party. For example, in Fig. 1, John has many different attributes. Each of the different identities John has with the persons and institutions John interacts with consists only of a subset of his attributes. Some identities can be linked because they share a unique attribute, e.g., his social security number across his healthcare-related identities. Other identities cannot be linked per se.

Managing one's identities means controlling which party should be privy of which attributes. A party can learn attributes of a user either because it performs its own identity vetting on the attributes (e.g., require the user to provide physical documents or take an exam), or because the users sends the party an attribute. To be guaranteed the correctness of an attribute in the latter case, a party (verifier) typically relies on another party (issuer) whose identity vetting procedures it trusts. In other words, a user needs to be able to transfer an attribute from one party, the issuer, to another party, the verifier, in a trusted way. The is the first basic mechanism that user-centric identity management requires. The second basic mechanism is one to (re-)authenticate users under a previously established identity. Both mechanisms can in principle be realized

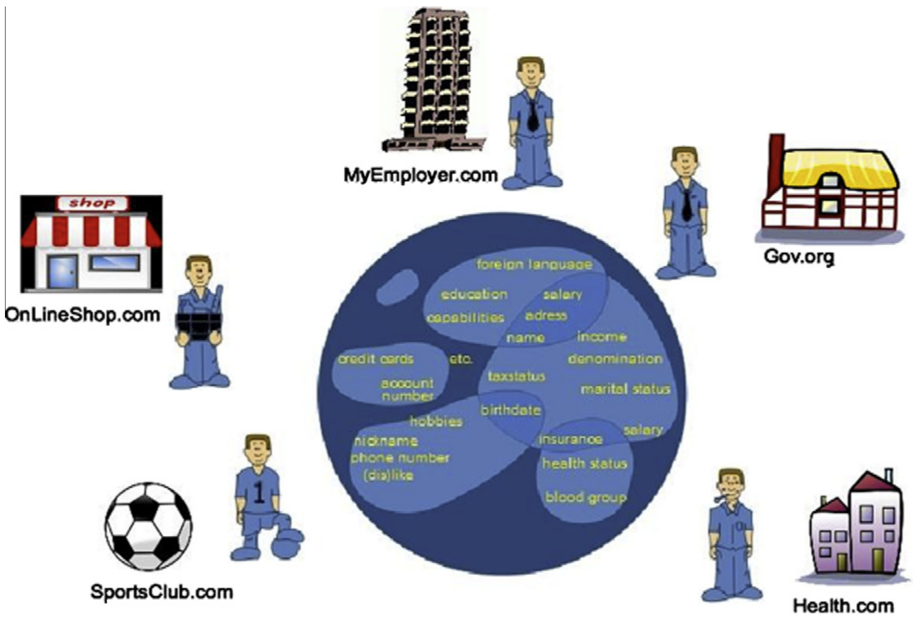

Fig. 1. John's multiple online identities [58,21].

based on simple username and password authentication, but this provides poor security guarantees. Indeed, passwords are well-known to be vulnerable to password guessing, phishing, and social engineering attacks. Their insecurity has an effect on privacy, too: to alleviate the shortcomings of password-based authentication, many service providers resort to collecting as much side information about the user as they can (e.g., location or transaction history) and analyzing that data to detect suspicious behavior and potential breaches. Thus, for security and privacy reasons, it seems advisable to use strong public-key cryptography for both mechanisms.

When realizing these two mechanisms for the digital world, it is not enough to mimic the paper-based solutions as is often done. Indeed, one has to consider the very different nature of the digital world where information is spread and processed so easily. One thus must adhere to the principle of data minimization: when a user transfers an attribute from an issuer to a relying party, neither party should be able to learn any information that is not already revealed by the transferred attribute, even if they collaborate. The classical approaches that are used today unfortunately do not satisfy this requirement. Private credentials [32,16,27] provide a solution that is very similar to the classical certificate-based approach in terms of the overall functionality and security guarantees, but at the same time allows the user to control and separate her different identities.

4.2. How private credentials work

Roughly speaking, private credentials follow the same approach as X.509 certificates. Each user generates a secret key and corresponding public key. The credential is a signature by the issuer on the user's attributes and her public key. To transfer attributes, the user signs a challenge message using her secret key and sends the signature along with her issuer-signed credential to the relying party. She can re-authenticate under a previously established public key by signing a challenge message using her secret key. However, private credentials have the following particular features:

Single secret key, many public keys: Instead of a single public key, each user can generate many public keys from her single secret key. These public keys cannot be linked to each other, i.e., given two public keys one cannot tell whether they belong to the same user or to two different users. Often these public keys are called (cryptographic) pseudonyms.

Transformable credentials: A credential issued to one public key can be (repeatedly) transformed into a credential that is valid on another public key of the same user. Essentially, the credential is thereby no longer bound to a unique public key of the user, but rather to the underlying secret key. Moreover, the transformed credential may contain a selected subset of the attributes contained in the original credential. Transformed credentials are unlinkable, meaning that for two transformed credentials with disjoint sets of revealed attributes, one cannot tell whether they originated from the same or from different credentials. Notice that all credentials, transformed as well as original ones, still verify correctly w.r.t. the issuer's verification key.

These properties are crucial to allow users to properly manage their different identities. Users can generate one public key for each of their identities. To transport attributes from an issuer to a relying party, the user first obtains a credential that includes those attributes for the public key by which she is known at the issuer. She then transforms the credential so that it contains only those attributes she wants to reveal and so that it is valid for the public key by which she is known at the relying party.

As one can see, the high-level principles of private credentials and traditional certificates are largely the same, the sole difference being that different cryptographic algorithms are used to generate public keys and to sign certificates and messages. Thus, private credentials can be used in any situation where traditional certificates can be used by simply replacing the algorithms. Private credentials provide the same level of security, but additionally guarantee privacy during the process.

Private credentials offer all features of traditional certificates, including revocation. They also offer a number of features that traditional cryptography cannot provide. For example, instead of revealing attribute values, the user can choose to merely reveal that some predicate over the attributes holds. In the identity card example above, Alice's transformed credential could for example reveal the statement that she is either German or French and that her birthday is before 1994, without revealing anything more about the exact value of her attributes.

Another feature is that private credentials allow users to provide some of the attributes in encrypted form. That is, if the relying party requires it, the user can encrypt an attribute (e.g., her name) under a trusted third party's public key (e.g., the police) and cryptographically prove to the relying party that the encryption contains her name as stated in her certified credential. When the user later abuses her anonymity (e.g., by misbehaving or causing damage), the relying party can ask the trusted third party to decrypt her true identity.

The fact that users can generate as many unlinkable public keys as they want can be undesirable in certain scenarios. For example, an online pollster may want to use private credentials so that eligible users can participate anonymously in the poll, but at the same time he does not want fanatic users to bias the result by participating multiple times. To prevent this, the pollster can optionally insist that the transformation of the public key is done in a particular, deterministic way, so that if the transformation is done a second time for the same poll, then the resulting public key will be the same. The pollster will then detect that this public key has already participated and refuse access. The user is of course aware of such imposed limitations, as they influence the credential transformation process.

4.3. Cryptographic realization

The probably most prominent realization of private credentials are IBM's Identity Mixer and Microsoft's U-Prove which are based on the Camenisch–Lysyanskaya credential system [27] and Brand's scheme [16], respectively. The implementation of the additional features described make use of cryptographic mechanisms such as verifiable encryption [30] and specific schemes for revocation such as dynamic accumulators [28] and proofs of list non-memberships [17,55]. The literature provides some alternative and more recent proposals (e.g., [8,1,28]), some offering additional features such as delegation of credentials or are proven secure under different cryptographic assumptions.

There are a number of concepts that are tightly related to private credentials. In fact, many of them can be extended into a private credential system. A group signature scheme [36,31,4,9,10,14] can be seen as a private credential system with only one issuer (the group manager) and the "group membership" as the sole attribute. Group signature schemes also include a dedicated authority called opening manager who can "open" the anonymity of a credential showing, i.e., is able to tell which member of the group did produce a particular signature. A blind signature scheme [33,45] allows a user to get a signature from the signer without the signer being aware of the message nor the resulting signatures. Thus they can also be seen as a private credential system with a single issuer, "possessing a credential" as the sole attribute, and where a credential can be used only once (otherwise the different uses would be linkable). Brands' credential scheme [16] is an extension of blind signatures, that includes attributes into signatures and where users get issued a whole set of signature that they can then use one by one.

Schemes such as e-cash [35] and direct anonymous attestation [19] are special cases of private credentials. E-cash is the digital equivalent of physical coins. A coin is essentially a private credential that can be spent only once. Different from physical coins, their cryptographic counterparts allow one to implement a number of restrictions on how coins can be spent [25]. For instance, to prevent money laundering one can specify that if user spends more money than a certain limit with a particular merchant then the user's identity is automatically registered (at the same time the identity is guaranteed to remain hidden if the limit is not surpassed).

Direct anonymous attestation [19] is a protocol that got standardized by the Trusted Computing Group. The scenario here is that (the owner of) a platform wants to prove to another entity that it is trusted, i.e., it is in a pristine state. For instance a user wants to prove to her bank that her computer is not compromised. To this end, the platform embeds a tamper-proof chip (called trusted platform module or TPM), that monitors the system and the software on the system and can then confirm to, e.g., the user's bank that the system has not been infected. To enable the bank to verify that claim, the TPM needs to authenticate itself as a valid TPM. Here is where private credentials come in: each TPM has its private credential embedded by its manufacturer. Now a TPM is able to authenticate itself as genuine without it being uniquely identified. To counter illegal cloning of TPMs (and the private credential imbedded into it), the direct anonymous attestation protocols include a mechanism to detect and reject cloned credentials.

4.4. Attribute based access control

Using private credentials for authentication and controlling access to resources requires a few changes to currently deployed systems. Today, authentication and access control is typically done by first identifying the user and then deciding whether or not the user is allowed access to the requested resources. A preferable approach would be to (1) define the access control policies in terms of what attributes (e.g., age, role, nationality, authorization, etc.) a user needs to possess to access a certain resource, then to (2) communicate the access control policy to the user and (3) allow the user to provide proof that she or he indeed possesses the necessary attributes.

Fig. 2 shows these flows and the possible components involved in this. We give a brief description here and refer to Bichsel et al. [13] for the full details. The server (e.g., service provider) has a repository of authentication policies stating for which resource it requires a user to present which attributes certified by which entity (identity provider). For example, a policy could specify that for accessing a chat-room, users need to be teenagers according to a national eID card. The user has a repository of credentials that she possesses. These credentials should preferably be private credentials as discussed before, but can also include for instance X.509 attribute certificates or links to OpenID or Facebook Connect attribute resources.

Now when a user want to access a resource or service, she sends a request to the server. Upon receiving an authentication request (1) the server determines and pre-evaluates the relevant authentication policy (1a) and sends it to the requesting user (2). During the pre-evaluation, variables in the policy such as the current date are resolved to generate the policy sent to the user. Having received the policy, the user's system determines which statements (i.e., *claims*) can be made based on the attributes contained in her credentials, such that the claims fulfill the received policy (2a). For example, a policy requiring one to be a teenager may be fulfilled based on a national ID card or a student ID. From all possible claims, the user can select (2b) which ones she indeed wants to submit to the server. The specific credential technologies are then instructed to transform the credentials into credentials that support the selected claims. When using private credentials, the transformed credentials (*evidence*) will only reveal the information specified by the selected claims (2c). If other credential technologies are used, more information is typically revealed. The selected claims are then sent together with the accompanying evidence to the server (3) who verifies that the claims imply the policy (3a), checks whether all claims are implied by the evidence, and that the evidence is valid (3b).

4.4.1. Policy languages

Realizing such an access control system requires the specification of the authentication policy for the server, the authentication policy that is sent to the user, and one for the claims that the user provides to the server. Then, formats of the cryptographic evidence need to be specified.

There is an OASIS standard "Identity Metasystem Interoperability" which defines the authentication policy and the claims the user sends to the server [49]. The discontinued Microsoft Cardspace was based on this standard. The standard is geared towards what attributes are requested from a user and hence does not allow one to express very simply attribute-based access control policies. Camenisch et al. [29] address these shortcomings and provide a credential-based authentication requirements language (CARL). However, CARL focuses only on the authentication policy that is sent to the user.

The ABC4Trust project provides a specification of all necessary data formats [26]. More precisely, XML specifications of all the objects that occur are given and then it is defined how these objects can be used to form authentication policies and claims together with the cryptographic evidence. The objects defined include pseudonyms, credentials, and public and secret keys of of credential issuers. Then the authentication policy is defined in terms of which credentials and pseudonyms to be presented. An authentication policy allows the specification of a number of conditions on the attributes contained in the different credentials. For instance it can be specified which attributes contained in a credential need to be revealed or that

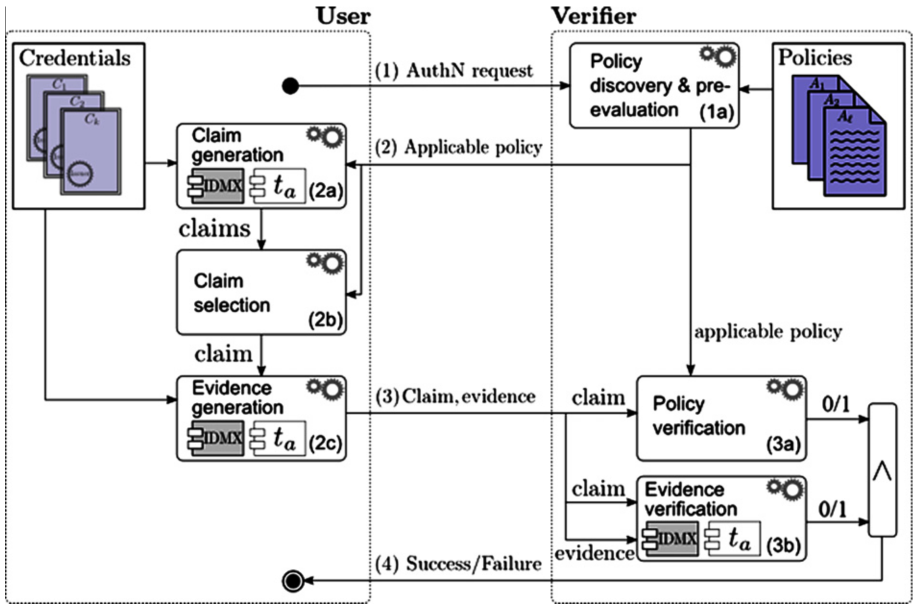

Fig. 2. Attribute based authentication flows and components [13].

a user must possess credentials from a number of issuers such that all these credentials where issued on the same last name (without the last name being revealed).

4.4.2. Implementation

CardSpace is Microsoft's implementation of some of the concepts we have discussed in this section. The research consortia ABC4Trust (www.abc4trust.eu) has put forth an architecture for privacy-protecting authentication [26]. The architecture unifies the different concepts and features of private authentication (including all the those discussed here), defines privacy-preserving attribute-based credentials (Privacy-ABCs), provides a language framework in XML schema, and presents the software components involved. We will discuss the language framework in more detail in Section 6. The project will publish a reference implementation of their architecture and is conducting two medium scale pilots to demonstrate the feasibility of private credentials in real world scenarios.

5. Data minimization at the application layer

Building an application that is data minimizing seems to be much harder than anonymous communication or privacy preserving authentication. Indeed, many applications are rather data maximizing – sometimes because of the requirements of the application (e.g., because the data can be sold to third parties) and sometimes because the implementation of the application is easier if more user data is available. For instance, it might be convenient to ask for the social security number just because it can serve as a unique database key for a user's record. Often applications require more data because they have been designed with standard authentication technologies in mind and just because of that require the fully identification of users.

Building a privacy–friendly application thus starts with making the business processes behind an application such that it needs only the minimally necessary data and then by using the right technologies to realize the application. Unfortunately, there is currently no tool-box available (other than what we have described in the preceding section) that could help with the technical realization of such applications. However, for a couple of scenarios there exist dedicated cryptographic protocols. In the following we describe them to give the reader a taste of what is possible to realize with cryptography.

5.1. Privacy-preserving access

In the preceding section we have discussed how to realize privacy preserving access control, i.e., where users requesting access to a resource need to reveal only the minimally necessary information about them. However, the party providing the resources still learns which resources the user at hand wants to access. We now discuss how the users can even hide which record they access.

As a motivating example, imagine a database containing valuable information and is therefore not sold as a whole, but rather customers are allowed access to records individually. Often, the list of queried records reveals sensitive information about the customers' intentions. For example, a company's search queries to a patent database or to a DNA genome database may reveal its research strategy or future product plans.

An oblivious transfer protocol [59] solves this apparently deadlocked situation by letting a client and server interact in such a way that the server does not learn anything about which record the client obtained, while the client can only learn the content of a single record.

Oblivious transfer, however, does not provide a full solution yet. For example, imagine that each record in a patent or DNA database as described above is protected by a different access control policy, describing the roles or attributes that a user needs to have in order to obtain it.

Fortunately, by combining anonymous credentials with adaptive oblivious transfer protocols, one can construct solutions where the user can obtain the records she is entitled to, without revealing the applicable access control policy to the database, or which roles she has [22]. By another combination of such techniques, the database can attach different prices for each record, and let users only download as many records as their prepaid balance allows, all the while remaining completely anonymous [23,61].

The idea underlying such protocols is as follows. The database server first encrypts each record with a unique key and publishes these encryptions. The encryption key is derived from the index of the record, the access control list of the record, and a secret of the database server. This is done such that, although the secret of the database is the same for all record keys, it is not possible to derive the encryption key for one record from that of another record. Thus, to decrypt a record the user needs to retrieve the corresponding key from the server. To be able to do this the user has to obtain necessary credentials from the issuer. Each anonymous credential [34,52,27], issued to a user, certifies the user's role or attribute (or whatever items the access control list requires). To obliviously access a record for which the user has the necessary credentials, she engages in a protocol with the database to retrieve the key. This protocol is an efficient and secure two-party computation to which the database inputs its secret and public keys and the user inputs the index of the records she wants to access, the access control list of that record, and her credentials. The two-party computation first verifies that the inputs of the user are all correct and valid. If this is the case, it outputs to the user the decryption key computed from the index of the record, the access control list of the record, and a secret of the database. Thus, if the protocol succeeds, the user can decrypt that record, otherwise, she cannot. The database learns nothing about the index of the record that is being accessed, nor about the categories associated with the record.

5.1.1. Electronic voting, polling, and petitions

Voting privacy is more than just a desirable feature, it is a fundamental principle for a democratic election. Electronic voting schemes have been proposed based on mix networks [32], based on homomorphic encryption [38], and based on blind signatures [46]. Electronic voting schemes form the backbone of e-democracy and should be properly designed and verified to guarantee a variety of security properties, such as end-to-end verifiability, voter anonymity, as well as coercion and receipt freeness.

Other mechanisms such as electronic petitions and verifiable electronic opinion polls aim at strengthening participatory democracy. As discussed in Section 4.2, private credentials have features that make them applicable to such scenarios. Several demonstrators have actually been built to show the feasibility of this approach [43,18,67,64]. The latter also implemented parts of the anonymous credential protocol in a chip card using software technology and hardware [56] similar to the one used in European identity cards. A deployment of such a system would bind an electronic petition signing directly to a European citizen. The properties of the anonymous credential system would allow to further restrict the scope of the partition. For instance for local issues it would be required to have residence in a particular district in order to be able to participate in the petition. Moreover, with this technology, it is simple to extend the application such that a poll may include restrictions on who can participate (e.g., only persons older than 18). Optionally, the user may selectively disclose attributes or properties of those attributes (e.g., an age interval) that may be used for statistics.

6. Controlling personal data released

So far we considered mechanisms to minimize the personal data that users have to reveal while at the same time achieving security. Even when these mechanisms are used, some personal data will often be revealed. Moreover, users want to tell others about them and share personal information such as blog posts, pictures, their current locations, etc.

Once released, data can only be controlled by technologies such as digital rights management (DRM). These technologies however have a number of short comings (e.g., it is virtually impossible to prevent “analogue” copies of the data being made and distributed) and also are not very suitable for the protection of personal data. Fortunately, such extreme protection as DRM seeks to provide is probably not necessary for personal information. More precisely, in a typical DRM scenario, the data to be protected is sent to a party one does not trust to either copy or distribute the data. In contrast, in the typical scenarios where users reveal personal data it seems very reasonable to assume that the party who receives data is not a copy pirate and can be trusted to protect the data sufficiently and to distribute it only to designed parties (if at all). So the goal here is rather to specify who is allowed to receive which data and to use it for what purpose.

It is useful to distinguish two cases according to whom users (want to) reveal personal data: (1) to other users such as family members, friends and colleagues and (2) organizations such as companies or government agencies. In the latter case, the data receiver will typically run an infrastructure to store the personal data and its business process will determine which data it needs. Thus, the data receiver can be expected to be able to tell the user what data it requires (authentication policy) for which purpose (data handling policy) and further to protect the received data adequately. In the former case, the data receiver will probably view the data with a web browser and might not store it, or if so, on a device with little protection in place. Also, here it is the users who need to define what data they want to send to whom and for what purpose. The means to enable a user to control the data released are quite different in the two cases. We discuss them in the next two subsections.

6.1. Revealing data to other users

Electronic networks have made it tremendously easy for users to communicate and to share data with each other. People send messages to each other, have their own web site where they present themselves, and interact with each other on social network sites such as Facebook and Linkedin.

When communicating like this, users typically intend to send their message (or picture, etc.) to a particular receiver or to a particular group of people. Despite this, users are not aware that their data will end up with many additional parties. For instance, email messages travel in the clear through a number of routers or profiles on an a social network site can typically be queried by third parties for all kinds of purposes. Unfortunately, users have little or no means to specify which parties they indend their data for, to ensure that solely the intended parties will receive their data, or to become aware of which additional parties will also be able to access the data.

6.1.1. Defining the audience

When for instance sending an email message, a user determines the audience of the message by entering the addressees in the relevant fields. However, when communicating via web sites, such social network sites, blogs, or fora, it is much harder for users to determine who should be able to receive or access the information they provide. Many sites such as blogs and fora do not put any restrictions at all on who can read information provided by users. Even if access is limited to registered users, contributing users can hardly tell who the registered users are. Social networks such as Facebook allow users some control on who can read their information. However, much profile information is public and moreover the controls offered to users are very coarse. For instance, allowing access to all ‘friends of friends’ might mean giving access to almost 17’000 people assuming an average of 130 friends – hardly a group of people one would have in mind when sending a message.

In PrimeLife [57,66], more versatile means to control the audience were investigated. For instance, van den Berg and Leenes have realized an experimental social network site called Clique [65] that allows users to define the audience of their profile information or postings. Instead of giving the users fixed categories such as ‘friends’ or ‘friends of friends,’ it allows users to define and name their own groups (called collections) from their contacts. When posting information to the site, a user is asked for which collections they intend the message. Once a target collection is selected, the user can still modify the audience by adding or excluding individual contacts (when discussing a present for a particular spare-time friend, one can thus chose to send the message to all spare-time friends except the prospective recipient of the present). Some of these concepts put forth in Clique are now found in the Google+ social network site.

Defining the audience by selecting a group of people is a relatively simple way of defining an access control policy. However, such a simple approach seems to work only for direct communication. Users might for instance not have all the intended recipients in their contact list, or might want to restrict the access by some condition such as a given time or specific roles such as the officer currently on duty or a member from a specific department. To this end, PrimeLife has investigated scenarios other than social networks where users reveal information to other users and studied what kind of access control policies users might want to create and how they can be enabled to do so.

One of the scenarios was the following which probably most of us can relate to. To protect personal information such as emails, family pictures, legal documents, and so on, it is best to properly encrypt them and back them up securely (best at a different place, possibly somewhere in the cloud). However, what happens if we loose the encryption keys, forget the passwords, are unconscious, or, in the worst case, die? We might want our relatives or friends to be able to access some of the data, depending on the circumstances. To address this, Borcea-Pfitzmann, Köpsell, Dobiáš, and Wahrig [15] have built a prototype that allows users to store their data encrypted on any of the popular cloud storage servers and then to delegate the right to access their data to other users. Thereby, users can first group their files into *areas of life*, and then select the delegatee(s), rights to and under what conditions. As delegatee individuals, a group of people or a role of a person can be defined. If the delegatee is a group of people, it can be specified what the quorum for obtaining access needs to be. As condition, a specific day, a time period, or conditions such as when in hospital, or when deceased is possible. It is also possible to revoke an access right later. The enforcement of the access control conditions in this prototype is mainly done via key management: the individual files are encrypted and then the keys stored at a trusted third party who enforces the access control policy. In practice, making such a strong use of a trusted third party would not be acceptable and other solutions to enforce the policies would be needed. This could include direct communication of the keys to the delegatees (assuming they all have public keys), or oblivious trusted third parties [24] who need to be only minimally trusted to perform a certain task such as for instance acting as time beacons.

6.1.2. Enforcing the audience

In the examples we discussed so far in this section, the access control policy defined by the user is enforced be a third party who hosts the data for the user (e.g., the social network site, an email provider). That however means that the data is also visible to these third parties. A preferable way to enforce the audience is by encrypting the data for the intended audience. This, however, requires that the respective public keys are available. Although no global public key infrastructure is available today, one can go quite a long way already, as we shall see.

One example is the Firefox plugin called Scramble! [62,7]. Developed for Facebook, Scramble allows one to replace any text in a web form by an encryption of that text under the public keys of the indented receivers. The selection process of the receivers is similar to the one developed by PrimeLife for its social network Clique that we described earlier. If the encryption does not fit the amount of text that the web form allows for, it will be replaced by a shortened URL that points to a third party server where the encryption is hosted. The encryption keys of the intended recipients are fetched from PGP key server. Also, the plugin can generate an encryption key pair if one does not have one and register the public key with one of the PGP key servers. Thus, if all one's friends are downloading the same plugin, they can exchange information via a social network provider without the provider learning the content of what is exchanged! A number of similar solutions have been proposed.

Another example of enforcing the audience is encrypting email. Most mail programs are able to sign and encrypt email. They will also automatically store the public keys of the sender who signed their emails.

Unfortunately, hardly anyone makes use of the available encryption technologies to protect their communicated data. One reason for this might be that users are offered little to no support in *managing* their keys and therefore do not dare to encrypt because loosing the decryption key (or deleting it because it expired) makes all data received from other users inaccessible.

6.1.3. Recognizing the real audience

Ideally, the audience one has defined matches the parties who become privy of the data one reveals. As we have argued already, today this is typically not the case. Even if the data is encrypted for the recipient(s), they might use it for purposes that the user is not aware of (in particular, if the recipients are organizations). So, applications should be built such that only the indended audience gets the data and if additional parties receive the data as well, then the user is made aware of this. We will discuss mechanisms for the latter in more detail in the next paragraph.

6.2. Revealing data to organizations

In revealing (personal) data to an organization, it is typically the organization who defines what data it requires from a user for what purpose. We call the statement of what data is required an authentication policy and the statement of for what purpose the data is used the data handling policy. The two together are often called privacy policy. Today these policies are often not stated explicitly. The data handling policy is often a site-wide legal text referred to by a link. The authentication policy is typically manifested by a registration page with a number of forms. Users can decide whether to consent to the data handling and releasing their data and to proceed or to abort the transaction. Sometimes a part of the data handling policy is also displayed on the registration page, e.g., users can opt-in or opt-out to receive a news letter or that their information be used for statistical purposes.

From a privacy point of view it is preferable that the authentication and data handling policy be explicitly communicated to the users and presented to them in an accessible way. We have already discussed how this can be done for authentication policies in Section 4.4. We now discuss the data handling policy.

6.2.1. Data handling policies

Today, the data handling policies of a website are usually communicated to the user via some link on that website that brings the user to a legal text describing what the website provider intends to do with the data received from the user. These texts are typically long and are formulated in a legal language which is hard to understand for the average user. So one can hardly argue that when they reveal their data, users are well informed as to what happens to their data. It is therefore preferable that users get some automated assistance to assess the data handling policy. One attempt to do so is the *Platform for Privacy Preferences (P3P)* [68]. It allows a website to express their data handling policy in a standardized format that can be automatically retrieved and interpreted and then displayed to the user. An example tool for this is the Privacy Bird [37,41], a browser plugin for Internet explorer. It can for instance warn users that a site aims to disclose their name and address to third parities. P3P policies could also be automatically compared with the preferences set by the user. To express preferences, the P3P Preference Exchange Language (APPEL) [40] has been put forth. Users for whom using the language is too complex could, instead of defining their own preferences, rely on templates provided, e.g., by data protection authorities. Unfortunately, P3P was never widely used and today many sites (including Google and Facebook) provide a dummy file instead of real P3P policy to satisfy browsers insisting on a P3P file.

P3P was criticized for being an isolated solution and not being enforceable either legally or technically. The former is because privacy legislation is quite different in many parts of the world while a P3P policy will be the same for all users of a website. The latter is because a P3P policy is not linked to a technical implementation ensuring that the policy will be enforced w.r.t. data received from a user.

To enable the (automatic) enforcement of a P3P policy, the enterprise privacy language (EPAL) [3] has been proposed. So while P3P expresses the privacy policy that an enterprise communicates externally to its customers, EPAL expresses the internal privacy (data handling) policies and practices. EPAL describes the policy that an enterprise enforces on the data it receives from users, i.e., it specifies who (e.g., sales department) is allowed to do what (e.g., store) what (e.g., customer address) for what (e.g., order-processing) under what conditions (e.g., user is older than 13 years of age) with what obligations (delete record after 3 months). Naturally, a P3P policy should be derivable from an EPAL policy for communication of the data handling policy to a user. Very often, a user's data does not stay within a single enterprise but often needs to be communicated to other enterprises to deliver a service to the user. Examples include payment (via a credit card company) and shipment (via a delivery service) of the goods sold by an enterprise. EPAL covers this as well, i.e., an EPAL policy is meant to stick to the received user data and be enforced whenever the data is accessed or used for some purpose. Also, when an enterprise sends a user's data to a second enterprise, it can be verified whether the second enterprise's EPAL policy allows for the enforcement of the EPAL policy the first enterprise has attached to the received user data. Although EPAL was submitted for standardization to W3C, it never found its way into practice.

The PrimeLife Policy Language (PPL) [2,20] developed by the PrimeLife project aims to replace P3P and EPAL, i.e., to provide a full framework for privacy aware access control. PPL extends XACML (extensible access control markup language), which is an OASIS standard for access control policies, in two ways. First, it extends XACML to enable credential based access control as described in Section 4.4. Second, it extends XACML to allow the specification of data handling policies. This latter extension not only allows enterprises to enforce their privacy policies internally but also to communicate the data handling policies to users as well as to other enterprises to which it sends the data. Also, the privacy preferences of users can be expressed by that extension – indeed there is no difference between users sending their data to an enterprise and an enterprise sending users' data to another enterprise. PPL allows for matching policies to see whether the policies of the one sending the data and the one receiving the data are compatible. As an XACML policy typically contains many references internal to an entity, a PPL policy gets sanitized before it is communicated to another party for matching. PrimeLife has implemented PPL and some components of this implementation are available from the project's website.

6.2.2. Enforcing data handling policies

For a company having received user data, a data handling policy essentially defines an access control policy to that data and enforcement requires similar components as a typical access control system with the following main differences. First, the access control enforcement points need to get more information about the entity who requests access to user data. That is, it not only needs to know the role or attribute of the entity (such as employee of marketing departments) but also the action the entity wants to take (read) and the purpose (to send marketing email). Second, data handling policies (should) contain obligations that need to be executed upon granting access to data or depending on some external conditions (e.g., delete after 3 months). Thus a special component is needed that manages and executes obligations. We refer for instance to Cassassa Mont and Thyne [54] and Trabelsi and Njeh [63] for details of systems for enforcing data handling policies.

Typically such access control enforcement systems assume an honest behavior of the companies receiving the data and of all their employees as the data are stored in the clear. This assumption can be weakened by using encryption for the data and using trusted hardware to implement the access control [39].

Another, at the moment still theoretical, approach is to employ attribute-based encryption or policy-based encryption [5,12]. Here, the data is encrypted in such a fashion that it can be decrypted only by entities who possess the required attributes. More precisely, there is a key distribution center who issues the entities the secret keys related to their attributes.

6.2.3. Authoring and displaying policies

Privacy policies at some point have to be authored and verified by humans. This requires specific tools that make this as easy as possible, in particular to end users who can not be expected to become IT specialists. While for the definition of users' privacy preferences one could resort to templates provided, e.g., by consumer organizations or privacy commissioners, the decision of whether or not to consent to a data handling policy cannot be delegated. Presenting a data handling policy to users such that they can quickly assess it is very challenging and far from solved.

One approach for this is by expressing the policy with icons rather than text. While they can legally not replace written text, they may supplement the text, e.g., pointing to relevant sections or by helping to access layered data handling policies. A number of proposals for such icons exist for different areas (c.f. Holtz et al. [48] and the references therein). Most of the proposals try to depict what a company aims to do with the data, such as using them for payment, deleting after them after 24 h, or for marketing. There are also some proposals for usage with user to user communication that try to capture what the user expects the recipient to do (not distribute, delete after reading, etc.).

7. Conclusion

We have argued that since Google, the Internet has become financed based on the collection and use of personal information. Because of the insecurity of the Internet, this puts the users' privacy and, probably more importantly, their security at risk and thus electronic commerce over the Internet. Now, this view is much too narrow. The Internet is rapidly expanding and becoming the communication media connecting all kinds of devices and networks, including personal smart phones and tables, sensors, and networks of large organizations. We are heading into a future where almost everything will have an IP address. The web will collect, process, and communicate loads of data, much of which will be personal or potentially sensitive. The security of this emerging infrastructure is of paramount importance and rather than patching later, security should be considered a design requirement. As it will be impossible to physically secure all this infrastructure, cryptographic mechanisms will have to be employed to encrypt and authenticate each and every bit, in a way such that each party becomes privy only to the information they indeed require and nothing more.

The mechanisms that we have discussed in this article will allow this to a large extent already. Still more research, both theoretical and very applied is needed to secure the emerging digital world. In the following, we discuss open research challenges we perceive and a roadmap to get the current state-of-the-art technologies into practice.

7.1. Roadmap

Proper security and privacy have to become core requirements for any mechanism or application that is built. When designing an application, the requirements specified are often a reflection of what the person(s) in charge understand of the available technologies and how they can be employed. This includes the way business processes are designed and how the low-level technical implementation is done. It is therefore essentials that the people in the whole design chain are aware of the possibilities of the state-of-the-art technologies, in particular the ones discussed in this article. The same is true for policy makers: to draft new regulations and laws to govern our digital environment, they need to be aware of the available technologies, their dangers and their merits, and how these can be used. Finally, to further foster market adoption the end-users need to be made aware of the risks of the current technologies and how these risks could be addressed with alternative or new technologies and mechanisms. Thus, the education about privacy-enhancing technologies is an essential step in the roadmap towards security of our future digital world.

Another important step is to make technologies such as those we have discussed easy to deploy and use. Indeed, a number of the technologies solve seemingly paradoxical problems and/or are quite complex and have a plethora of different features. Therefore, they are not straightforward to employ and use. To overcome this hurdle, their application programming interfaces (APIs) need to be made as simple as feasible, possibly using reasonable defaults. Also, the relevant APIs and communication formats need to be standardized, whenever feasible by extending existing standards. This process is best accompanied by real-world pilot deployments to validate the results.

Finally, it will be necessary to analyze different application scenarios and show alternative realizations that are more secure and better protect the privacy of the users. This most likely will require new technologies to be developed or at least require the innovative use of existing ones. This will demonstrate the feasibility of such approaches on the one hand and drive research and innovation on the other hand.

7.2. Challenges

There is a considerable number of open problems and challenges that need to be addressed towards realizing a secure digital environments.

Currently large parts of the Internet are financed by using on personal information, most prominently to provide targeted advertisement. So, on the one hand, means need to be found to allow the use of personal information without the information being spread. For instance, the collection and processing of the information could be done in a distributed fashion or even locally on user's devices. On the other hand, new economic models and incentives have to be researched to make data parsimonic applications attractive or at least viable.

A whole set of challenges evolves around the users. The burden to manage and secure their data is put upon them. This includes data such as usage profiles, credit cards data, pictures, or personal documents. Users need support in managing these data, in being able to judge where they are stored and processed and what risks this involves, and in making informed decisions. Security of the users' data will require that these data be encrypted and that strong authentication mechanisms be used. This means that the user further needs support in managing their cryptographic keys and credentials. Today, no usable tools are available to users to safely back up, recover if lost, or refresh their key materials!

7.3. Towards a safe digital society

Our society is shaped by the technologies we use. This is not a new phenomenon; it has probably always been like this. As an example, consider cars and transportation in general. The availability of relatively affordable transportation has had a big impact on where we live, where we work, on our whole lifestyle. The more refined and sophisticated the technologies become, the harder it is to understand them and to judge how and to what extent (if at all) these technologies should be used and what impact that will have.

The digital world is expanding quickly into all aspects of the physical world. It probably has never been easier to introduce new technologies, and get them used by people all over the world. Many of these technologies have a great potential to change our world for the better – but also for the worse. Scientists and engineers as well as the companies inventing, building, and introducing these technologies all alike need to take up their responsibility to fully understand the good as well as the bad potentials of the technologies, initiate discussions about them, and, last but certainly not least, to explore and provide means to make the future digital society a safe and pleasant place.

Acknowledgements

The author had the pleasure to work with many fine people on privacy enhancing mechanisms including Endre Bangerter, Patrik Bichsel, Maria Dubovitskaya, Robert Enderlein, Susan Hohenberger, Markulf Kohlweiss, Anja Lehmann, Anna Lysyanskaya, Joy Müller, Gregory Neven, Franz-Stefan Preiss, Victor Shoup, Dieter Sommer, and Abhi Shelat and many colleagues and friends from the EC-funded projects PRIME, PrimeLife, and ABC4Trust. Günter Karjoth, Claudia Diaz, George Danezis, Stefan Köpsell, and Harry Rudin were so kind to comment on drafts of this manuscript. Thanks to all of you!

References

[1] Norio Akagi, Yoshifumi Manabe, Tatsuaki Okamoto, An efficient anonymous credential system, in: Gene Tsudik (Ed.), Financial Cryptography, Lecture Notes in Computer Science, vol. 5143, Springer, 2008, pp. 272–286.

[2] C.A. Ardagna, S. Capitani Di Vimercati, G. Neven, S. Paraboschi, E. Pedrini, F.S. Preiss, P. Samarati, M. Verdicchio, Advances in access control policies, Privacy and Identity Management for Life (2011) 327–341.

[3] P. Ashley, S. Hada, G. Karjoth, C. Powers, M. Schunter, Enterprise privacy authorization language (EPAL). Research report, 3485, 2003.

[4] Giuseppe Ateniese, Jan Camenisch, Marc Joye, Gene Tsudik, A practical and provably secure coalition-resistant group signature scheme, in: Mihir Bellare (Ed.), Advances in Cryptology—CRYPTO 2000, Lecture Notes in Computer Science, vol. 1880, Springer Verlag, 2000, pp. 255–270.

[5] Walid Bagga, Refik Molva, Policy-based cryptography and applications, in: Andrew S. Patrick, Moti Yung (Eds.), Financial Cryptography, Lecture Notes in Computer Science, vol. 3570, Springer, 2005, pp. 72–87.

[6] BBC, Privacy backlash over girls around me mobile app, April 2012. <http://www.bbc.co.uk/news/technology-17582975>.

[7] Filipe Beato, Markulf Kohlweiss, Karel Wouters, Scramble! your social network data, in: Simone Fischer-Hübner, Nicholas Hopper, (Eds.), PETS, Lecture Notes in Computer Science, vol. 6794, 2011, pp. 211–225.

[8] Mira Belenkiy, Jan Camenisch, Melissa Chase, Markulf Kohlweiss, Anna Lysyanskaya, Hovav Shacham, Randomizable proofs and delegatable anonymous credentials, in: Shai Halevi (Ed.), CRYPTO, 2009.

[9] Mihir Bellare, Daniele Micciancio, Bogdan Warinschi, Foundations of group signatures: formal definitions, simplified requirements, and a construction based on general assumptions, in: Eli Biham (Ed.), Advances in Cryptology—EUROCRYPT 2003, Lecture Notes in Computer Science, vol. 2656, Springer Verlag, 2003, pp. 614–629.

[10] Mihir Bellare, Haixia Shi, Chong Zhang, Foundations of group signatures: the case of dynamic groups, in: Alfred Menezes (Ed.), Topics in Cryptology—CT-RSA 2005, Lecture Notes in Computer Science, vol. 3376, Springer, 2005, pp. 136–153.

[11] Oliver Berthold, Hannes Federrath, Stefan Köpsell, Web mixes: a system for anonymous and unobservable internet access, in: Workshop on Design Issues in Anonymity and Unobservability, Lecture Notes in Computer Science, vol. 2009, Springer, 2001, pp. 115–129.

[12] J. Bethencourt, A. Sahai, B. Waters, Ciphertext-policy attribute-based encryption, in: IEEE Symposium on Security and Privacy, 2007 (SP'07), IEEE, 2007, pp. 321–334.

[13] Patrik Bichsel, Jan Camenisch, Franz-Stefan Preiss, A comprehensive framework enabling data-minimizing authentication, in: Proceedings of the 7th ACM Workshop on Digital Identity Management, ACM, 2011, pp. 13–22.

[14] Dan Boneh, Xavier Boyen, Short signatures without random oracles, in: Christian Cachin, Jan Camenisch (Eds.), Advances in Cryptology—EUROCRYPT 2004, Lecture Notes in Computer Science, vol. 3024, Springer, 2004, pp. 54–73.

[15] Katrin Borcea-Ptzmann, Stefan Köpsell, Jaromir Dobiáš, Hagen Wahrig, D1.3.2 – prototype of a selected scenario on privacy throughout life, 2011. <http://www.primelife.eu/results/documents/142-132d>.

[16] Stefan Brands, Rethinking Public Key Infrastructure and Digital Certificates—Building in Privacy. PhD thesis, Eindhoven Institute of Technology, Eindhoven, The Netherlands, 1999.

[17] Stefan Brands, Liesje Demuynck, Bart De Decker, A practical system for globally revoking the unlinkable pseudonyms of unknown users, in: Josef Pieprzyk, Hossein Ghodosi, Ed Dawson (Eds.), ACISP, Lecture Notes in Computer Science, vol. 4586, Springer, 2007, pp. 400–415.

[18] Stefan Brands, Christian Paquin, E-participation proof of concept with u-prove technology. online, 2010. <http://www.microsoft.com/mscorp/twc/endtoendtrust/vision/eid.aspx> (accessed in November 2010).

[19] Ernie Brickell, Jan Camenisch, Liqun Chen, Direct anonymous attestation, in: Proceedings of the 11th ACM Conference on Computer and Communications Security, ACM Press, 2004, pp. 225–234.

[20] Laurent Bussard, Gregory Neven, Franz-Stefan Preiss, Matching privacy policies and preferences: access control, obligations, authorization, and downstream usage, Privacy and Identity Management for Life (2011) 311–326.

[21] J. Camenisch, A. Lehmann, G. Neven, Electronic identities need private credentials, Security & Privacy, IEEE 10 (1) (2012) 80–83.

[22] Jan Camenisch, Maria Dubovitskaya, Gregory Neven, Oblivious transfer with access control, in: Ehab Al-Shaer, Somesh Jha, Angelos D. Keromytis (Eds.), ACM Conference on Computer and Communications Security, ACM, 2009, pp. 131–140.

[23] Jan Camenisch, Maria Dubovitskaya, Gregory Neven, Unlinkable priced oblivious transfer with rechargeable wallets, in: Financial Cryptography 2010, 2010.
[24] Jan Camenisch, Thomas Groß, Thomas S. Heydt-Benjamin, Accountable privacy supporting services, Identity in the Information Society (2009).
[25] Jan Camenisch, Susan Hohenberger, Anna Lysyanskaya, Balancing accountability and privacy using e-cash (extended abstract), in: SCN, Lecture Notes in Computer Science, vol. 4116, 2006, pp. 141–155.
[26] Jan Camenisch, Ioannis Krontiris, Anja Lehmann, Gregory Neven, Christian Paquin, Kai Rannenberg, Harald Zwingelberg, Architecture for attribute-based credential technologies version 1, 2011. <https://abc4trust.eu/index.php/pub/results/107-d21architecturev1>.
[27] Jan Camenisch, Anna Lysyanskaya, Efficient non-transferable anonymous multi-show credential system with optional anonymity revocation, in: Birgit Pfitzmann (Ed.), Advances in Cryptology—EUROCRYPT 2001, Lecture Notes in Computer Science, vol. 2045, Springer Verlag, 2001, pp. 93–118.
[28] Jan Camenisch, Anna Lysyanskaya, Signature schemes and anonymous credentials from bilinear maps, in: Matthew K. Franklin (Ed.), Advances in Cryptology—CRYPTO 2004, Lecture Notes in Computer Science, vol. 3152, Springer Verlag, 2004, pp. 56–72.
[29] Jan Camenisch, Sebastian Mödersheim, Gregory Neven, Franz-Stefan Preiss, Dieter Sommer, A language enabling privacy-preserving access control, in: SACMAT 2010, ACM, 2010.
[30] Jan Camenisch, Victor Shoup, Practical verifiable encryption and decryption of discrete logarithms, in: Dan Boneh (Ed.), Advances in Cryptology—CRYPTO 2003, Lecture Notes in Computer Science, vol. 2729, 2003, pp. 126–144.
[31] Jan Camenisch, Markus Stadler, Efficient group signature schemes for large groups, in: Burt Kaliski (Ed.), Advances in Cryptology—CRYPTO '97, Lecture Notes in Computer Science, vol. 1296, Springer Verlag, 1997, pp. 410–424.
[32] David Chaum, Untraceable electronic mail, return addresses, and digital pseudonyms, Communications of the ACM 24 (2) (1981) 84–88.
[33] David Chaum, Blind signatures for untraceable payments, in: David Chaum, Ro nald L. Rivest, Alan T. Sherman (Eds.), Advances in Cryptology—Proceedings of CRYPTO '82, Plenum Press, 1983, pp. 199–203.
[34] David Chaum, Security without identification: transaction systems to make big brother obsolete, Communications of the ACM 28 (10) (1985) 1030–1044.
[35] David Chaum, Amos Fiat, Moni Naor, Untraceable electronic cash, in: Shafi Goldwasser (Ed.), Advances in Cryptology—CRYPTO '88, Lecture Notes in Computer Science, vol. 403, Springer Verlag, 1990, pp. 319–327.
[36] David Chaum, Eugène van Heyst, Group signatures, in: Donald W. Davies (Ed.), Advances in Cryptology—EUROCRYPT '91, Lecture Notes in Computer Science, vol. 547, Springer-Verlag, 1991, pp. 257–265.
[37] CMU Usable Privacy and Security Laboratory, Privacy Bird: Find websites that respect your privacy. <http://www.privacybird.org/>.
[38] Josh D. Cohen, Michael J. Fischer, A robust and verifiable cryptographically secure election scheme (extended abstract), in: FOCS 1985, IEEE, 1985, pp. 372–382.
[39] Stephen Crane, Siani Pearson, Security/trustworthiness assessment of platforms, in: Jan Camenisch, Ronald Leenes, Dieter Sommer (Eds.), Digital Privacy – PRIME, Lecture Notes in Computer Science, vol. 6545, Springer, 2011, pp. 457–483.
[40] L. Cranor, M. Langheinrich, M. Marchiori, A p3p preference exchange language 1.0 (appel 1.0), W3C working draft, 15, 2002.
[41] L.F. Cranor, P. Guduru, M. Arjula, User interfaces for privacy agents, ACM Transactions on Computer–Human Interaction (TOCHI) 13 (2) (2006) 135–178.
[42] George Danezis, Claudia Diaz, Paul F. Syverson, Systems for anonymous communication, in: B. Rosenberg, D. Stinson (Eds.), CRC Handbook of Financial Cryptography and Security, CRC Cryptography and Network Security Series, Chapman & Hall, 2010, pp. 341–390.
[43] Claudia Diaz, Eleni Kosta, Hannelore Dekeyser, Markulf Kohlweiss, Girma Nigusse, Privacy preserving electronic petitions, Identity in the Information Society 1 (1) (2009) 203–209.
[44] Roger Dingledine, Nick Mathewson, Paul F. Syverson, Tor: the second-generation onion router, in: USENIX Security Symposium, USENIX, 2004, pp. 303–320.
[45] Atsushi Fujioka, Tatsuaki Okamoto, Kazuo Ohta, Interactive bi-proof systems and undeniable signature schemes, in: Donald W. Davi es (Ed.), Advances in Cryptology—EUROCRYPT '91, Lecture Notes in Computer Science, vol. 547, Springer-Verlag, 1991, pp. 243–256.
[46] Atsushi Fujioka, Tatsuaki Okamoto, Kazuo Ohta, A practical secret voting scheme for large scale elections, in: Jennifer Seberry, Yuliang Zheng (Eds.), Advances in Cryptology—AUSCRYPT '92, Lecture Notes in Computer Science, vol. 718, Springer, 1992, pp. 244–251.
[47] Jens Groth, A verifiable secret shuffle of homomorphic encryptions, in: Public Key Cryptography, Lecture Notes in Computer Science, vol. 2567, Springer, 2003, pp. 145–160.
[48] Leif-Erik Holtz, Harald Zwingelberg, Marit Hansen, Privacy policy icons, in: Jan Camenisch, Simone Fischer-Hübner, Kai Rannenberg (Eds.), Privacy and Identity Management for Life, Springer, Berlin Heidelberg, 2011, pp. 279–285.
[49] Michael B. Jones, Michael McIntosh, Identity metasystem interoperability version 1.0, July 2009. <http://docs.oasis-open.org/imi/identity/v1.0/identity.html>.
[50] T. Kohno, A. Broido, K.C. Claffy, Remote physical device fingerprinting, IEEE Transactions on Dependable and Secure Computing 2 (2) (2005) 93–108.
[51] Xiangfang Li, Lijun Qian, J. Kamto, Secure anonymous routing in wireless mesh networks, in: International Conference on E-Business and Information System Security, 2009 (EBISS '09), May 2009, pp. 1–5.
[52] Anna Lysyanskaya, Ron Rivest, Amit Sahai, Stefan Wolf, Pseudonym systems, in: Howard Heys, Carlisle Adams (Eds.), Selected Areas in Cryptography, Lecture Notes in Computer Science, vol. 1758, Springer Verlag, 1999.
[53] Prateek Mittal, Nikita Borisov, Shadowwalker: peer-to-peer anonymous communication using redundant structured topologies, in: Proceedings of the 2009 ACM Conference on Computer and Communications Security, CCS 2009, Chicago, IL, USA, November 9–13, 2009, ACM, 2009.
[54] Marco Casassa Mont, Robert Thyne, Privacy policy enforcement in enterprises with identity management solutions, Journal of Computer Security 16 (2) (2008) 133–163.
[55] Toru Nakanishi, Hiroki Fujii, Yuta Hira, Nobuo Funabiki, Revocable group signature schemes with constant costs for signing and verifying, in: Stanislaw Jarecki, Gene Tsudik (Eds.), Public Key Cryptography, Lecture Notes in Computer Science, vol. 5443, Springer, 2009, pp. 463–480.
[56] Oracle, Java card technology, 2012. <http://www.oracle.com/technetwork/java/javacard/overview/index.html> (accessed in September 2012).
[57] PrimeLife project. <www.primelife.eu>.
[58] Stefanie Pötzsch, Katja Liesebach, Hilko Donker, Privacy and identity management for europe (prime) – general public tutorial, February 2008. <https://www.prime-project.eu/tutorials/gpto/index_html/document_view>.
[59] Michael O. Rabin, How to exchange secrets by oblivious transfer, Technical Report TR-81, Harvard Aiken Computation Laboratory, 1981.
[60] Jian Ren, Jie Wu, Survey on anonymous communications in computer networks, Computer Communications 33 (4) (2010) 420–431.
[61] Alfredo Rial, Markulf Kohlweiss, Bart Preneel, Universally composable adaptive priced oblivious transfer, in: Pairing '09: Proceedings of the 3rd International Conference Palo Alto on Pairing-Based Cryptography, Springer-Verlag, Berlin, Heidelberg, 2009, pp. 231–247.
[62] Scramle! firefox plugin website. <http://homes.esat.kuleuven.be/fbeato/extras/scramble>.
[63] Slim Trabelsi, Akram Njeh, Policy implementation in XACML, in: Jan Camenisch, Simone Fischer-Hübner, Kai Rannenberg (Eds.), Privacy and Identity Management for Life, Springer, Berlin Heidelberg, 2011, pp. 355–374.
[64] Carmela Troncoso, Josep Balasch, Claudia Diaz, Venelin Gornishki, Markulf Kohlweiss, Michal Sterckx, Victor Sucasas, Adapid d15: e-government ii, Adapid deliverable, 2010.
[65] Bibi van den Berg, Ronald Leenes, Keeping up appearances: audience segregation in social network sites, in: Serge Gutwirth, Yves Poullet, Paul De Hert, Ronald Leenes (Eds.), Computers, Privacy and Data Protection, Springer, 2011, pp. 211–231.
[66] Bibi van den Berg, Stefanie Pötzsch, Ronald Leenes, Katrin Borcea-Pfitzmann, Filipe Beato, Privacy in social software, in: Jan

Camenisch, Simone Fischer-Hübner, Kai Rannenberg (Eds.), Privacy and Identity Management for Life, Springer, Berlin Heidelberg, 2006, pp. 33–60.

[67] Kristof Verslype, Jorn Lapon, Pieter Verhaeghe, Vincent Naessens, Bart De Decker, PetAnon: a privacy-preserving e-petition system based on Idemix, CW Reports CW522, Department of Computer Science, K.U. Leuven, October 2008.

[68] W3C, The Platform for Privacy Preferences 1.1 (P3P1.1) Specification, 2006. <http://www.w3.org/TR/P3P11/>.

[69] Douglas Wikström, Jens Groth, An adaptively secure mix-net without erasures, in: ICALP, Lecture Notes in Computer Science, vol. 4052, Springer, 2006, pp. 276–287.

Jan Camenisch received a Diploma in Electrical Engineering in 1993 and a Ph.D. in Computer Science in 1998 both from ETH Zurich. From 1998 until 1999 he has been Research Assistant Professor in Computer Science at the University of Aarhus, Denmark. Since 1999 he is Research Staff Member and project leader at IBM Research – Zurich. He was also the technical leader of the EU-funded projects PRIME (prime-project.eu) and PrimeLife (primelife.eu) which both contributed towards making on-line privacy a reality. His research interests include public key cryptography; cryptographic protocols, in particular those supporting privacy and anonymity; practical secure distributed computation; and privacy-enhancing technologies.