# *An Entity-centric Approach for Privacy and Identity Management in Cloud Computing*

Pelin Angin, Bharat Bhargava, Rohit Ranchal, Noopur Singh
Department of Computer Science
Purdue University
West Lafayette, IN, USA
{pangin, bbshail, rranchal, singh91}@purdue.edu

Lotfi Ben Othmane, Leszek Lilien
Department of Computer Science
Western Michigan University
Kalamazoo, MI, USA
{lotfi.benothmane, leszek.lilien}@wmich.edu

Mark Linderman
Air Force Research Laboratory
Rome, NY, USA
mark.linderman@rl.af.mil

***Abstract*—Entities (e.g., users, services) have to authenticate themselves to service providers (SPs) in order to use their services. An entity provides personally identifiable information (PII) that uniquely identifies it to an SP.**

**In the traditional application-centric Identity Management (IDM) model, each application keeps trace of identities of the entities that use it. In cloud computing, entities may have multiple accounts associated with different SPs, or one SP. Sharing PIIs of the same entity across services along with associated attributes can lead to mapping of PIIs to the entity.**

**We propose an entity-centric approach for IDM in the cloud. The approach is based on: (1) active bundles—each including a payload of PII, privacy policies and a virtual machine that enforces the policies and uses a set of protection mechanisms to protect themselves; (2) anonymous identification to mediate interactions between the entity and cloud services using entity's privacy policies.**

**The main characteristics of the approach are: it is independent of third party, gives minimum information to the SP and provides ability to use identity data on untrusted hosts.**

***Keywords-active bundles; cloud computing; identity management (IDM); personally identifiable information (PII); anonymous identification; zero-knowledge proofs (ZKP); privacy-enhancing technologies (PET); privacy; security.***

## I. INTRODUCTION

### A. Identity Management

An *identity* is a set of unique characteristics of an entity: an individual, a subject, or an object. An identity used for identification purposes is called an *identifier* [1].

Entity identifiers are used for authentication to service providers (SPs). Identifiers provide assurance to an SP about the entity's identity, which helps the SP to decide whether to permit the entity to use a service or not.

Entities may have multiple digital identities. An *Identity Management System* (*IDM*) supports the management of these multiple digital identities. It also decides how to best disclose PII to obtain a particular service. IDM performs the following tasks [2]:

1) *Establish identities:* Associate PII with an entity.
2) *Describe identities:* Assign attributes identifying an entity.
3) *Record the use of identity data:* Log identity activity in a system and/or provides access to the logs.
4) *Destroy an identity:* Assign expiration date to PII. PII become unusable after the expiration date.

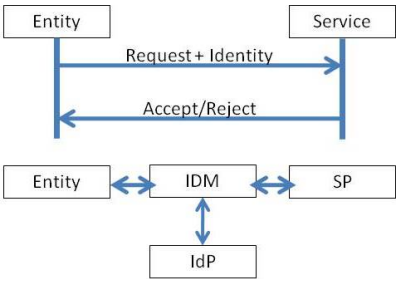


Figure 1. Authentication using Third Party Identity Management

Fig. 1 shows an example of authentication that uses PII. A user wants to use a service, for which she needs to authenticate to the SP but does not want to disclose her identity data. She has to disclose PII to uniquely identify herself to the SP. The main problem is to decide which information she should disclose and how to disclose it.

A set of parties use IDM and collaborate to identify an entity. These parties are (cf. [3]):

1) *Identity provider (IdP).* It issues digital identities. For instance, credit card providers issue identities enabling payment, governments issue identities to citizens.
2) *Service provider (SP).* It provides services to entities that have required identities. For instance, a user needs to provide identity information to SP to file her tax online.
3) *Entity.* Entities about whom *claims* are made. A claim could be, for example, a name, or a date of birth, etc.
4) *Identity verifier.* It receives requests from SP for verifing a claim about a specific entity. It verifies the correctness and decides whether the claim is correct or not.

An IDM uses one of the following three categories of identifiers: (1) information that both an entity and SP know, such as passwords; (2) information that an entity knows and SP can verify that IdP approved it, such as Social Security Number; and (3) information about the entity, such as fingerprints.

### B. Privacy in Cloud Computing

*Privacy* in cloud computing is the ability of a user or a business to control what information they reveal about themselves over the cloud or to a cloud service provider, and the ability to control who can access that information. Numerous existing privacy laws impose the standards for the collection, maintenance, use, and disclosure of personal information that

must be satisfied by cloud providers. The nature of cloud computing has significant implications for the privacy of personal, business and governmental information. Cloud SPs can store information at multiple locations or outsource it, then it is very difficult to determine, how secure it is and who has access to it [4].

A cloud SP is a third party that maintains information about, or on behalf of, another entity. Whenever an individual, a business, a government agency, or other entity shares information in the cloud, privacy or confidentiality questions may arise [4]. Trusting a third party requires taking the risk of assuming that the trusted third party *will* act as it is expected (which may not be true all the time). Cloud Computing spawns huge risks such as: (i) expected losses from a single breach can be significantly large; and (ii) the heterogeneity of "users" represents an opportunity of multiple collaborative threats. The main problems associated with such a model are:

1) *Loss of control*: Data, applications, and resources are located with SP. The cloud handles IDM as well as user access control rules, security policies and enforcement. The user has to rely on the provider to ensure data security and privacy, resource availability, monitoring of services and resources.
2) *Lack of trust:* Trusting a third party requires taking risks. Basically trust and risk are opposite sides of the same coin. Some monitoring or auditing capabilities would be required to increase the level of trust.
3) *Multi-tenancy:* Tenants share resources and may have opposing goals which could be conflicting. There is a need to provide a degree of separation between tenants.

### C. Identity Management in Cloud Computing

In the traditional application-centric [5] IDM model, each application keeps trace of entities that uses it. In cloud computing, entities may have multiple accounts associated with different SPs. Also, entities may use multiple services offered by the same SP (e.g., Gmail and Google Docs are offered by Google). A cloud user has to provide his personally identifiable information (PII), which identifies him while requesting services from the cloud. This leaves a trail of PII that can be used to uniquely identify, or locate a single entity, which—if not properly protected—may be exploited and abused. Sharing PIIs of the same entity across services along with associated attributes can lead to mapping of PIIs to the entity [5]. The main issue is how to secure PII from being used by unauthorized parties in order to prevent serious crimes against privacy, such as identity theft [4].

The owner of data, including PII, is responsible for his privacy in cloud computing. The owner needs technical controls supporting this challenging task.

Cloud computing requires an entity-centric model where every entity's request for any service is bundled with the entity's identity and entitlement information [6]. We propose an *entity-centric IDM* that allows the entity: (1) to create and manage its *digital identities* to authenticate in a way that does not reveal its *actual identities* or relationships between identities to vendors, service providers, etc; and (2) to protect

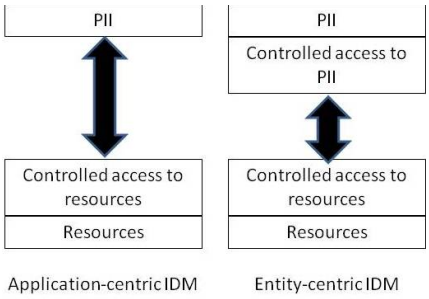


Figure 2. Application-centric vs Entity-centric IDM

PII from unauthorized access. Fig. 2 compares application-centric and entity-centric IDMs. The advantage of an entity-centric IDM is that a disclosure of PII is no longer arbitrarily or at the will of SP but is at the will of its owner. In this paper, we propose an approach for designing an entity-centric IDM model.

### D. Contribution and Paper Organization

We propose an approach for IDM in the cloud that: (1) does not use trusted third parties; and (2) can be used on untrusted or unknown hosts. The approach is based on: (1) active bundles—each including PII, privacy policies and a virtual machine that enforces the policies and uses a set of protection mechanisms (such as integrity checks, apoptosis, evaporation, decoy) to protect themselves; and (2) anonymous identification—to mediate interactions between an entity and cloud services using entity's privacy policies.

The paper is organized as follows: Section 2 discusses related work. Section 3 describes the Active Bundle scheme. Section 4 presents our proposed approach for protecting PII in cloud computing. Section 5 presents a sample scenario for the use of our approach. Section 6 concludes the paper.

## II. Related Work

This section discusses three known solutions for IDM.

### A. PRIME

Privacy and Identity Management for Europe (PRIME) [7] provides privacy-preserving authentication using anonymous credentials. The user-side component uses protocols for getting third party (IdP) endorsements for claims to *relying parties* (RPs). Anonymous credentials are provided using an identity mixer protocol (based on the selective disclosure protocol) that allows users to selectively reveal any of their attributes in credentials obtained from IdP, without revealing any of their information. The credentials are then digitally signed using a public key infrastructure. A major limitation of PRIME is that it requires both user agents and SPs to implement the PRIME middleware, which hinders standardization.

### B. Windows CardSpace

Windows CardSpace [8] is a plug-in for Internet Explorer 7, in which every digital identity is a security token. A security token consists of a set of claims, such as a username, a

user's full name, address, SSN etc. The tokens prove that the claims belong to the user who is presenting them.

When a CardSpace-enabled application or website wishes to authenticate a user, it requests a particular set of claims from the user. The user selects an InfoCard to use among the ones visually presented to him, and the CardSpace software contacts an IdP to obtain a digitally signed XML token that contains the requested information, which is communicated to the requesting application.

The CardSpace framework is criticized due to its reliance on the user's judgment of the trustworthiness of an RP. Most users do not pay attention when asked to approve a digital certificate of an RP, either because they do not understand the importance of the approval decision or because they know that they must approve the certificate in order to get access to a particular website. RPs without any certificates at all can be used in the CardSpace framework (given user consent). Even if an RP presents a higher-assurance certificate, the user still needs to rely on an IdP providing that certificate to the RP, thus the user needs to trust the IdP. Another drawback is that, in a case where a single IdP and multiple RPs are involved in a single working session, (which we expect to be a typical scenario) the security identity metasystem within the session will rely on a single layer of authentication, that is, the authentication of the user to the IdP. If a working session is hijacked or the password is cracked the security of the entire system is compromised.

### C. Open ID

OpenID [9] is a decentralized authentication protocol that helps cloud users in managing their multiple digital identities with greater control over sharing of their PII. A user has to remember one username and password—an OpenID. She can log onto websites with this OpenID. She interacts with an RP that provides means to specify an OpenID for the authentication. The user has previously registered an OpenID with an OpenID provider (a TTP). Upon being discovered by the RP, the OpenID provider authenticates (commonly by prompting a password) and asks the user whether the RP should be trusted to receive the necessary identity details for the service. If she accepts, she is redirected to the RP along with her credentials, which need to be confirmed by the RP to provide service. After the OpenID has been verified, authentication is considered successful, and the user is considered logged in to the RP under the identity specified by the given OpenID.

OpenID has been termed "phishing heaven" due to its susceptibility to phishing attacks and social engineering. A malicious attack can be easily set up to lure users into entering their authentication information at a website that poses as an OpenID provider website [10, 18].

## III. Active Bundle Scheme

### A. Overview of the Active Bundle Scheme

An *active bundle* (*AB*) is a container with a payload of sensitive data, metadata, and a virtual machine (VM) [11]. *Sensitive data* constitutes content to be protected from privacy violations, data leaks, unauthorized dissemination, etc. *Metadata* describes the active bundle and its privacy policies. The metadata includes the following components (details available in [11], [12]): (a) *provenance* metadata; (b) *integrity check* metadata; (c) *access control* metadata; (d) *dissemination control* metadata; (e) *life duration* value; (f) *security* metadata (including: *security server id*; *encryption algorithm* used by the VM; *encrypted pseudo-random number generator*; *trust server id* used to validate the trust level and the role of a host; and *trust level threshold* required to access data in an active bundle); and (g) other application-dependent and context-dependent metadata. The *virtual machine (VM)* manages and controls the program code enclosed in a bundle. The main VM functions include (a) enforcing bundle access control policies through *apoptosis*, *evaporation*, or *decoy* actions (e.g., disclosing to a guardian only the portion of data that the guardian is entitled to access); (b) enforcing bundle dissemination policies; and (c) validating bundle integrity.

Although one of the goals of ABs is to protect sensitive data from unauthorized disclosure by malicious hosts, the scheme has its own threat model. The main element is that it requires a correct execution of VM by hosts.

### B. Prototype of Active Bundle Using Mobile Agents

We have developed a prototype in the context of mobile agent paradigm using TTPs. The prototype was developed using the Java mobile agent framework JADE [14].

#### B.1. Description of the ABTTP Architecture

The system contains: AB Coordinator (ABC); AB Destination; Directory Facilitator (DF); AB Services including: Security Services Agent (SSA), Trust Evaluation Agent (TEA), and Audit Services Agents (ASA); and an AB.

The components are distributed among 4 containers[1].

*B.1.1. Active Bundle Coordinator*: *ABC* hosts the Jade *Directory Facilitator*, providing a yellow pages[2] service. It is used by agents to register and deregister their services, to modify their registered description, and to search for services. In our prototype we register four agents: AB, SSA, TEA and ASA. They communicate even when they are on different hosts.

*B.1.2. Client Application*: It includes ABC, which hosts *Container 1*. ABC accepts from a user input including sensitive data, metadata, and the new destination for the AB. Then, it creates an AB and gives it the input of the user. The AB transforms the user input to its own attributes, which compose the AB's structure. It also includes a set of functions that compose the AB's virtual machine. Next, the ABC registers the AB in the DF.

*B.1.3.Active Bundle Destination*: ABD is an application that hosts a container. The only function of this component is receiving active bundles.

---

[1] A container is a Java process. It should not be confused with a host. A host may run one or many containers at the same time.

[2] A yellow page is a registry of entries which associate service descriptions to agent IDs [14].

*B.1.4. Active Bundle Services*: They include three agents: SSA, TEA, and ASA. The first agent, SSA, maintains a database of information about ABs. This information is used for encrypting and decrypting sensitive data and metadata included in ABs. Each AB is described in SSA using the following information: name, decryption key, and the threshold trust level that a host must satisfy to use the AB. SSA stores the identity data of the AB in file on the SSA host. The second agent, TEA, answers requests from SSA about the trust level of a specified host, which could be obtained using a trust management system [15]. The third agent, ASA, monitors activities of ABs. It receives audit information from ABs, and records this information into a file for analysis by authorized entities[3] (e.g., AB owners, or auditors).

*B.1.5. Active Bundles*: An AB is a mobile agent that has a set of attributes and operations. It is constructed by an ABC (details in Subsection B.2.2). The constructed AB provides its owner[4] with a list of ABDs (destinations), and asks to select one as its destination. Upon receiving the destination choice, the AB starts preparing itself for the move. This step is called *building ABs* (details in Subsection B.2.3). Upon arriving at the destination, the AB enables itself (details in Subsection B.2.4).

*B.2. Behavior of the Active Bundle Components*

This section describes the behavior of ABs as a sequence of initialization, building, and enabling steps that followed.

*B.2.1. Initialization of an AB*: An owner of sensitive data provides ABC with sensitive data and metadata, including access control and dissemination control metadata. ABC constructs an AB by putting together data, metadata, and adding a virtual machine. After this stage, the AB becomes an active entity (since it has its own virtual machine) that can perform the remaining steps of this algorithm.

*B.2.2. Building an AB*: The steps are:

1) The AB gets from SSA two pairs of public/private keys where the first pair of keys is used for encrypting the AB and the second pair of keys is used for signing/verifying the signature of sensitive data included in the AB. The reason for having two key pairs is to prevent attackers from modifying AB's sensitive data and signing it again with the public key of the data owner.
2) The AB sends a request to SSA asking it to record the AB's security information. The AB's identity data includes its name, a decryption key, and the trust level that a host must satisfy to use the AB. The goal is to keep the decryption keys and other auxiliary data for ABs in a trusted location. The decryption keys are given only to hosts that are eligible[5] to access the AB.
3) The AB computes a hash value for sensitive data and signs them using the signature key. The signature certifies that sensitive data is from its owner.
4) The AB encrypts sensitive data using the encryption key.

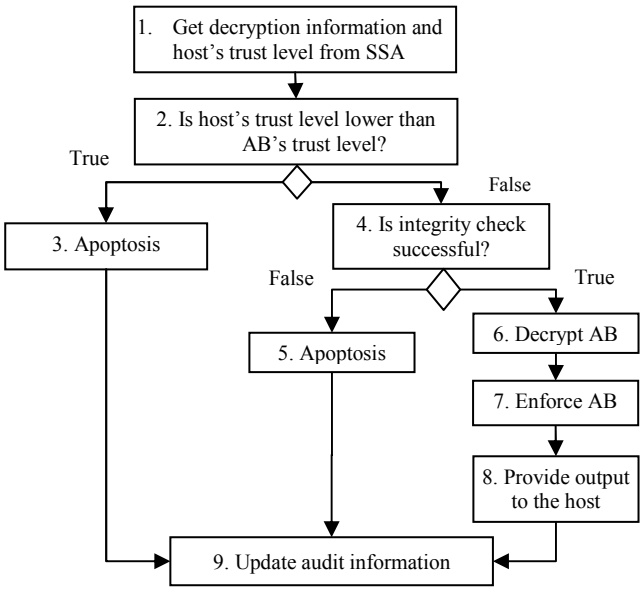


Figure 3. A UML activity diagram for enabling an active bundle

*B.2.3. Enabling of an active bundle*: After arriving at the destination host, the active bundle enables itself (cf. Fig. 3). The steps of the enabling algorithm are as follows:
Step 1: AB sends a request to SSA asking for the security information on AB and the host's trust level.
Step 2: AB checks if the host's trust level is lower than the minimal trust level required for AB access. If yes, the AB apoptosizes (executes Step 3); otherwise, it executes Step 4.
Step 4: AB checks integrity of AB's sensitive data. It computes the hash value for sensitive data. Then, it verifies the AB's signed hash value by comparing it to the computed hash value. If the verification fails, AB apoptosizes (Step 5); otherwise, the AB decrypts its sensitive data (Step 6).
Step 7: AB enforces its privacy policies.
Step 8: AB provides the output to the host.
Step 9: AB sends audit information to ASA. This information includes AB's name, the host's identity, and the name of the event being audited ("the move to the host").

## IV. PROPOSED APPROACH FOR PROTECTING PII IN CLOUD COMPUTING

This section describes the proposed approach for an entity-centric IDM model using the Active Bundle scheme and the anonymous identification method.

### A. Characteristics of the Existing Approches

These solutions have two *characteristics*, which are:

1. *The use of Trusted Third Party*. The major issues for adopting such an approach for cloud computing are: (i) the trusted third party (it could be a cloud service located at the cloud provider) and the service provider may be the same. Therefore the trusted third party may not be an independent-trusted entity anymore; (ii) it is a

[3] Note that this prototype does not include audit analysis tools. This feature is one of our future works.

[4] One of our future tasks is to improve the prototype such that a recipient of an active bundle can disseminates it further.

[5] By eligible we mean that the visited host has enough trust level that allows it to access some or all sensitive data included in the active bundle.

centralized approach. But if the Trusted Third Party is compromised; the PII of its users is compromised too.

2. *They do not support untrusted hosts.* The client application that holds the PII must be executed on a trusted host such that the host does not extract the PII.

### B. Selected Research Problems

The research problems are:

1) *Authenticate without disclosing data (unencrypted data):* When a user sends the identity information to get authenticated for a service, it may encrypt the data. However, before this information is used by the service provider, it is decrypted such that the service provider can use it. But as soon as the data is decrypted it become prone to attacks. This is particularly of a concern if the provider decides to store this data.
2) *Use service on untrusted hosts (hosts not owned by user):* The available IDM solutions need the user to be on a trusted host for using the IDM system or service. They do not allow usage of IDM on untrusted hosts like public host. With the advances in cloud computing where data may reside anywhere in the cloud, this issue needs to be addressed.
3) *Minimal disclosure and minimize risk of disclosure during communication between user and service provider (protect from Side Channel and Correlation Attacks):* Data needs to be protected from disclosure. In the scenario of cloud computing, this becomes even more important where the sensitive data may be held by a service provider and it is transmitted to another service provider (as a subcontractor), to use the service.

### C. Approach

We propose an approach for entity-centric IDM based on the use of AB scheme to protect PII from untrusted hosts, which we name it *IDM Wallet*. We use Zero-knowledge proof for authentication of an entity without disclosing its identifier, which we name *anonymous identification*. Fig. 4 shows the structure of IDM Wallet and anonymous identification.

With Anonymous identification, it is possible to prove a claim or assertion (authenticate) without actually disclosing any credentials. However, consider a case in which a user buys books from Amazon. The user needs to provide his address to receive the books by mail. There are situations where multiple parties are involved in the same transaction and need different information from the user. The shipping company needs to know the address, Amazon should not learn her address, but wants to be sure that user gives a real address to the shipping company. In this case IDM Wallet, after Anonymous identification, creates an AB that includes the PII (address in this case) that needs to be disclosed. This AB is a token that includes metadata, access control policies, and VM, besides the PII (here it is only the address). This token is given to SP who can give it to the mailing company. Using IDM wallet gives us protection to use on untrusted hosts and sending token as AB protects the PII when disseminated to SP.

### D. Description of IDM Wallet

An IDM Wallet is an AB, which holds user identities and manages their disclosure. It has the following structure (as seen in Fig.4):

1) *Identity data:* The data used during authentication, getting service, using service (i.e. SSN, Date of Birth). This data is encrypted and enclosed inside the IDM Wallet.
2) *Disclosure policy:* This is a set of rules for choosing Identity data from a set of identities in IDM Wallet. For instance, if a particular identity data has been used for a particular service then the same data needs to be used and disclosed every time for the same service. There is no need to disclose another item of PII to that service.
3) *Disclosure history:* This can be used for logging and auditing purposes and selecting the Identity data to be disclosed based on previous disclosures.
4) *Negotiation policy:* This is Anonymous Identification, based on the Zero Knowledge Proofing. It is described in the next subsection.
5) *Virtual Machine:* This contains the code for protecting PII data on untrusted hosts. It enforces the disclosure policies.

### E. Description of Anonymous Identification.

We describe Fiat and Shamir identification and signature scheme [16]. We discuss the use of the scheme for identification.

#### E.1. Description of Fiat-Shamir Identification Scheme

The goal of Fiat and Shamir identification scheme is to allow SP to verify the PII of an entity. The scheme prevents SP from using the PII of the customer entity to identify itself [16]. The scheme has two protocols: issuing identity to an entity, and verifying identity of an entity. Before issuing an identity, IdP chooses a public integer $n$ and a pseudo random

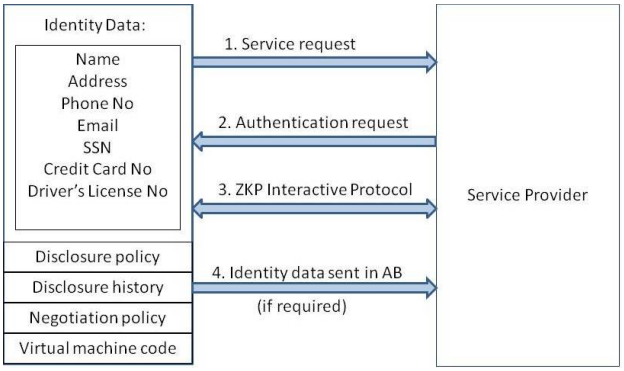


Figure 4. Structure of IDM Wallet

function $f$ where $f$ associates arbitrary strings to elements of the range $[0,n)$. Integer $n$ is the product of two secret prime numbers $p$ and $q$.

*Protocol 1*(issuing identities by an IdP to an Entity): IdP checks the physical identity of an entity and prepares a string I, which contains all the relevant information about the entity (It also includes the validity of the identity (expiration date,

limitations on validity, etc). The IdP then performs the following steps:

i. Compute values $v_j = f(I, j)$ for information at indices $j$.
ii. Pick $k$ distinct values of $j$ for which $v_j$ is a quadratic residue (mod $n$) and compute the smallest square root which contains $I$, the $k$ $s_j$ values, and their indices.
iii. Issue an identity, which contains $I$, $k$ $s_j$ values and the selected k indices. (To simplify notation we assume that the first $k$ indices $j = 1, 2,\ldots, k$ are used.)

*Protocol 2 (verifying the identity of the entity):* Assuming SP has the universal modulus $n$ and function $f$. The goal of this step is that the entity (Party A) proves to SP (Party B) that it is the owner of PII $I$. The steps of the protocol are:

i. A sends $I$ to B.
ii. B generates $v_j = f(I, j)$ for $j = 1,\ldots, k$.

Repeat steps (iii) to (vi) for $i = 1,\ldots, t$

iii. A picks a random $r_i \in [0,n)$ and sends $x^i = r_i^2 \pmod n$ to B.
iv. B sends a random binary vector $(e_{i1},\ldots, e_{ik})$ to A.
v. A sends to B: $y_i = r_i \prod_{eij=1} s_j \pmod n$.
vi. B checks that $x_i = y_i^2 \prod_{eij=1} v_i \pmod n$.

In Fiat and Shamir scheme, the entity (Party A) gives the information $I$ to the SP (Party B). This allows B to verify that IdP issues $I$ for A. Since A does not know function $f$ then B knows at the end of the protocol that $I$ indeed identifies A, if all the $t$ checks are successful.

*E.2. Sketch of the Proposed Anonymous Identification Sheme.*

In the following, we adapt Fiat and Shamir identification scheme [16] such that Party A does not give information $I$ to Party B (as in step $i$ of *Protocol 2*). The protocol verifies anonymously the membership of a value in a set of values.

In the following we give a sketch of the new protocol which allows SP to verify the identity of entities without knowing the entities' PII. The scheme includes two protocols. We use *Protocol 1* of Fiat and Shamir as is and we change *Protocol 2*. The main steps of *Protocol 2* are:

i. IdP sends to B: $v_j = f(I, j)$ for $j = 1,\ldots, n$.

Repeat steps (ii) to (v) for $i = 1,\ldots, t$

ii. A picks a random $r_i \in [0,n)$ and sends $x^i = r_i^2 \pmod n$ to B.
iii. B sends a random binary vector $(e_{i1},\ldots, e_{ik})$ to A.
iv. A sends to B: $y_i = r_i \prod_{eij=1} s_j \pmod n$.
v. B checks that $x_i = y_i^2 \prod_{eij=1} v_j \pmod n$.

In this protocol SP holds all possible values of an attribute. The role of A is to match one of its information with one of the legitimate values that SP holds.

*F. Simulating the Use of the Proposed Approach for Entity-centric IDM*

The following is a scenario simulating the use of the proposed approach:

1) An entity requests a service from an SP (e.g., a website).
2) In response to the service request, the SP informs the entity, through a technical policy requirement, that in order to gain access to the service the user must authenticate and provide its identity information (if required).
3) IDM Wallet uses its technical policy and determines what claims it should issue given the proof supplied by the claimant and the technical policy requirements of the SP.
4) IDM Wallet, as instructed by the entity satisfies the technical policy requirements of the SP and runs an interactive protocol based on *Anonymous Identification* with the SP for authentication.
5) If further required by SP, IDM Wallet creates an AB token and forwards it to the SP. SP then uses its technical policy to decide whether to recognize (authenticate) the user and provide the service.

*G. Characteristics and advantages of the Proposed IDM*

The characteristics of the proposed approach are:

1) *Ability to use Identity data on untrusted hosts.* It has a self-integrity check to find if the data is tampered. If the integrity is compromised, it will destroy the data itself by doing apoptosis or evaporation to protect it from falling into wrong hands.
2) *Independent of third party*. This prevents correlation attacks, man in the middle attacks, side-channel attacks and protects from the problem of third party being compromised, since the exchange of data from AB to host is local to the host. It increases the trust by putting the user in control of who has his data and how it is being used in the process of authentication, negotiation, and data exchange.

Advantages of the proposed approach include the following:

1) *Independent and trustworthy*. since the interaction is only between the SP and the user,
2) *Gives minimum information to the SP*. SP receives only necessary inforormation.
3) *Portability*. IDM Wallet can be carried on mobile, or flash drive etc.

## V. Sample Application: Privacy for Blind

In the world of growing security and privacy concerns, the blind and visually impaired people are even more vulnerable than others. Navigation aids—such as the white cane—cause the ones who use them be easily recognized by others around them, including malicious people, posing security threats.

A recent proposal for the design of a navigation system for the blind and visually impaired [17] uses the power of mobile and cloud computing to provide context-awareness. The proposed architecture has two main components, which are a mobile device with integrated location sensing module responsible for local navigation, local obstacle detection and avoidance, as well as interacting with the user and the cloud side, and the Web Services Platform employed to support functionalities including outdoor navigation, indoor navigation and object recognition.

Location tracking is an essential component of any context-aware system, as the location of a user/device provides a wealth of clues about the immediate surroundings. There-

fore, location-based services among others would be the most frequently utilized type of Web services in the system proposed for independent navigation of the blind and the visually impaired.

When submitting their location information to the cloud, a blind user (and, in fact, any other user) could have security concerns that a malicious party could use this information to locate the user and harm or exploit the user for his own benefit. Therefore, any location data submitted to the cloud for a service invocation should not be linkable to the identity of the user of the service. The identity management system we propose in this paper ensures that the privacy of the user of a location-based service is preserved by following the minimal data disclosure principle as well as destruction of information by the active bundle upon meeting service providers with unacceptable trust levels.

We describe how the IDM system proposed handles different scenarios encountered while communicating with the provider of a route planning service for pedestrians.

The digital identity of the user stored in the active bundle in this case consists of sensitive information including name, home address, phone number, emergency contact etc. as well as a subscriber ID for the route planning service. Following the minimal data disclosure principle, the active bundle is programmed to only disclose the subscriber ID and the location of the user through evaporation of the remaining data upon arriving at the service provider, as the remaining information is not needed for this type of service. Before the active bundle from the mobile device of the user discloses the required information to the route planning service provider SP with a request for guidance from point X to point Y, it checks the trust level of SP using a trust server. If the trust level is below the specified threshold, the active bundle is destroyed with the apoptosis function, therefore preserving the privacy of all information in the bundle.

If SP passes the trust level check, the active bundle proceeds to the authentication phase. Disclosing the subscriber ID and the current location of the user at this step would violate location privacy of the user, as the subscriber ID is part of PII. What the proposed system does instead is to authenticate the user using the Zero-knowledge proof in the virtual machine on the active bundle only with the subscriber ID. Using this mechanism, the user is proven to be a subscriber of the service without disclosing the actual value of the identity. Thus, the location information cannot be linked to the identity of the user, preserving location privacy. This approach also makes inference of long-term behavioral patterns (such as frequently visited locations) much harder since the identity of a user is not linkable to the user's invocations of cloud services.

## VI. Conclusion

With the immense growth in the popularity of cloud computing, privacy and security have become a critical concern for both the public and private sector. There is a strong need for an efficient and effective privacy-preserving system based on entity-centric approach. The system should: (1) be independent of any trusted third party; (2) be able to unambiguously identify users that can be trusted both across the Web and within enterprises; and (3) be able to protect users' Personally Identifiable Information (PII).

Identity management (IDM) is one of the core components for cloud privacy and security. Cloud computing can benefit from the entity-centric mechanism for protecting privacy of sensitive data throughout their entire lifecycle. This mechanism, known as the *active bundle scheme*, was presented. It is able to provide users with control over their data, allowing them to decide what and when data will be shared.

The future work will involve development of a prototype of the proposed system for cloud computing and testing it for diverse real-world scenarios. The goal is to prove effectiveness of the proposed privacy and identity management system, as well as its potential for becoming a standard for privacy and identity management in cloud computing.

## VII. Acknowledgement

This research was supported by a contract from AFRL and NGC Corp. The authors at Purdue University are listed alphabetically by their last names.

## References

[1] A. Josang and S. Pope. User Centric Identity Management, In Proc. AusCERT, Gold Coast, May 2005.

[2] Wikipedia. Identity Management Systems. July 2010. http://en.wikipedia.org/wiki/Identity_management_systems

[3] K. Cameron and M.B. Jones. Design Rationale behind the Identity Metasystem Architecture. January 2006. http://research.microsoft.com/enus/um/people/mbj/papers

[4] R. Gellman. Privacy in the Clouds: Risks to Privacy and Confidentiality from Cloud. World Privacy Forum, 2009.

[5] A. Gopalakrishnan. Cloud Computing Identity Management. SETLabs Briefings, Vol 7, 2009.

[6] Identity Theft Primer, Libery Alliance Whitepaper, http://www.projectliberty.org/, December 05, 2005.

[7] S. Hubner. PRIME, https://www.prime-project.eu/. 2010.

[8] W. Alrodhan and C. Mitchell. Improving the Security of CardSpace, EURASIP Journal on Info Security Vol. 2009.

[9] OPENID, http://openid.net/, 2010.

[10] K. Cameron, Identity Weblog. 2010. http://www.identityblog.com/?p=685

[11] L. Ben-Othmane and L. Lilien. Protecting Privacy in Sensitive Data Dissemination with Active Bundles. Proc. 7th Annual Conference on Privacy, Security & Trust (PST 2009), Saint John, New Brunswick, Canada, August 2009.

[12] L. Lilien and B. Bhargava. A Scheme for Privacy-preserving Data Dissemination. IEEE Trans. on Systems, Man and Cybernetics, Part A: Systems and Humans, 2006.

[13] L. Ben-Othmane. "Protecting Sensitive Data During Their Life Cycle," Ph.D. Thesis, Western Michigan University, 2010 (in preparation).

[14] F. L. Bellifemine, G. Caire and D. Greenwood. Developing Multi-Agent Systems with JADE, John Wiley & Sons Ltd, West Sussex, England, 2007.

[15] Y. Zhong and B. Bhargava. Using Entropy to Tradeoff Privacy and Trust. SKM, Amherst, NY, Sep. 2004.

[16] A. Fiat and A. Shamir. How to prove Yourself: Practical Solutions to Identification and Signature Problems. CRYPTO, 1986.

[17] P. Angin, B. Bhargava and S. Helal. A Mobile Cloud Collaborative Traffic Lights Detector for Blind Navigation. 1st MDM International Workshop on Mobile Cloud. 2010.

[18] C. Sample and D. Kelley. Cloud Computing Security: Routing and DNS Threats. 2009. http://www.securitycurve.com/wordpress/