Western Michigan University

The Hilltop Review

Volume 3
Issue 1 *Fall*

Article 4

October 2009

# Privacy-preserving Transactions on the Web

Sahil Behl
*Western Michigan University*

Leszek T. Lilien
*Western Michigan University*

# PRIVACY-PRESERVING TRANSACTIONS ON THE WEB

***By Sahil Behl and Leszek Lilien, Ph.D.***
Department of Computer Science
College of Engineering and Applied Sciences

**Abstract.** There is a rapid growth in the number of applications using sensitive and personal information on the World Wide Web. This growth creates an urgent need to maintain the anonymity of the participants in many web transactions and to preserve the privacy of their sensitive data during data dissemination over the web. First, maintaining the anonymity of users on the World Wide Web is essential for a number of web applications. Anonymity cannot be assured by single interested individuals or an organization but requires participation from other web nodes owned by other entities. Second, preserving the privacy of sensitive data is another very important issue in web transactions. Today, exchanging and sharing personal data between various participants in web transactions endangers privacy. In this article, we discuss various research directions and challenges that need to be addressed while trying to accomplish our goal of maintaining the anonymity of participants and preserving the privacy of sensitive data in web transactions. To maintain anonymity of participants in a web transaction, we propose a method based on the modified form of the *club mechanism with economic incentives*, a solution which rests upon the Prisoner's Dilemma approach. We compare our approach to other well-known data-sharing approaches such as Crowds, Tor, Tarzan and LPWA. To maintain the privacy of sensitive data, we propose a solution based on privacy-preserving data dissemination (P2D2). We also present a solution to implement our approach using Semantic Web Rule Languages and Jena—a Java-based inference engine.

## 1. Introduction

Through a broad range of devices such as computers, personal digital assistants (PDAs), cell phones, and other web-enabled devices, the World Wide Web is now reaching the widest audience ever. There is growth in the number of computer applications that use sensitive and personal information such as medical data, credit card numbers, and other personally identifiable information. This creates an urgent need to maintain the anonymity of the participants in web transactions and to preserve the privacy of their sensitive data during data dissemination over the web. To maintain the *anonymity* of the communicating parties, we must make the source and destination of the data untraceable. We can define the *privacy* of a party as the capability to hide sensitive data from entities that are not entitled to view it.

* Accepted for publication on April 24, 2007

Maintaining the anonymity of users on the World Wide Web is essential in a number of web applications. For example, some users may want to participate in a Usenet discussion without revealing their identities. Some examples of such discussions include: (a) a discussion list for patients with sensitive diseases like AIDS, (b) the exchange of politically or socially subversive ideas, and (c) the expression of an opinion, such as a comment about one's supervisor, which may have repercussions if the identity of the author is revealed [Dura03].

Anonymity cannot be created by a single interested individual or by an organization, but requires participation from other web nodes owned by other entities. The more nodes participating in the mixing of the traffic, the better the anonymity, but establishing and maintaining trust among a large number of nodes can be a major impediment to sustaining such a framework. Each node is dependent on other nodes for protecting its anonymity, and hence, an appropriate economic incentive could be one of the solutions for managing distributed trust in such a framework [JeLB04].

Preserving privacy of sensitive data is another very important issue in web transactions. Today, exchanging and sharing personal data between various participants in web transactions endangers privacy. For example, in the healthcare domain, the age and sex of a patient and the month of discharge from the hospital are sufficient to identify the patient in a limited population. Likewise, knowing two childbirth dates is enough to identify one woman in a sizeable population [KDTC04].

Typical web transactions are two-party transactions. The *strength* of a party in a transaction is defined as the capability to demand private information from another party and the enforcements available when the other party refuses to comply [JeLB04].

To maintain anonymity of participants in a web transaction, we propose a method based on the modified form of the *club mechanism with economic incentives* [JeLB04], a solution which rests upon the Prisoner's Dilemma approach. We compare our approach to other well-known data-sharing approaches such as Crowds [ReRu97], TOR [TorO06], Tarzan [FrMo02] and Lucent Personalized Web Assistant (LPWA) [GGKM99].

To maintain the privacy of sensitive data, we propose a solution based on *privacy-preserving data dissemination* (P2D2) [LiBh06]. We also present a solution to implement our approach using Semantic Web Languages and a Java-based inference engine supporting Semantic Web Languages.

The rest of this article is broadly divided as follows: in Section 2, we discuss anonymizing participants in web transactions in general; in Section 3, we describe preserving the privacy of sensitive data in web transactions; in Section 4, we discuss various existing methods and our proposed solution for anonymizing participants in web transactions; in Section 5, we discuss our proposed solution for preserving privacy of sensitive data in web transactions, and in Section 6, we present our conclusion.

# 2. Anonymizing Participants in Web Transactions

## 2.1. Introduction

This section provides a brief explanation for the need to anonymize web transactions. An analysis of traffic over the web provides valuable information about the participants in web transactions. This information includes the IP address from which the participant's geographical location can be determined. Furthermore, a lot about the habits of the participant can be deduced by tracking the sites she visits frequently, the number of messages sent or received during the day, and with whom the participant interacts the most on the web and at what time of the day she usually browses the web.

Maintaining the anonymity of participants in web transactions is one of the greatest challenges for researchers today. The failure of a commercial solution—Freedom Networks initiated by Zero Knowledge Systems—further raises a question about this scenario [JeLB04]. The designers of this network admit that the network failed because the company could not sell its services to a sufficient number of clients to cover its costs.

This section describes the existing *club mechanism with economic incentives* [JeLB04] and its drawbacks, and the alternate design approaches that we considered to maintain the anonymity of the participants in web transaction.

## 2.2. Club Mechanism with Economic Incentives

In this section, we address a method to anonymize web transactions using club mechanism with economic incentives. The method uses an economic scheme in which each participant has to pay the central authority a one-time initiation fee and fines for misbehavior. In the model, each web transaction is considered as a *Prisoner's Dilemma* where two players have the option of cooperating or defecting while maintaining each other's anonymity. Table 1 shows the Prisoner's Dilemma in which the agent can either defect (D) or cooperate (C).

Table 1. The Prisoner's Dilemma game used in the club mechanism with economic incentives (cf. [JeLB04]).

| | **C** | **D** |
|---|---|---|
| **C** | $P_t, P_t$ | $-P_t, P_t + l_t$ |
| **D** | $P_t + l_t, -P_t$ | $-P_t + l_t, -P_t + l_t$ |

Let $P_t$ be the benefit from the *privacy protection* received by an agent within the time period *t*. Therefore, $P_t$ is the *cost of privacy violation* if it is suffered by a violation by an agent within the time period t. Also, $l_t$ is the benefit from *disclosing the privacy of another agent* within time period t. The assumption made is that the benefits gained from privacy protection are higher than the benefits received by sacrificing

the partner's privacy (i.e., $P_t > l_t$). Also, both parties in the web transaction have symmetric privacy needs.

The club mechanism with economic incentives consists of a Central Repository (CR) that randomly matches any two nodes (or club members) to perform a web transaction. Each club member is called an *agent*. During a web transaction, each agent has the option of either cooperating or defecting. If the original sender of the message feels that the intermediate agents have cheated on him by revealing his anonymity, then he reports it to the CR. If the CR discovers the fraudulent agent, the agent has to pay a fine to the original sender of the message whose privacy has been violated.

The next section describes various design approaches for anonymizing web transactions.

## 2.3. Alternative Designs for Anonymizing Web Transactions

In this section, we discuss the use of our design for a bidding application. This is only an example since our design can be used for other web applications as well. A *bidding system* is one in which each bidder places a bid for the product on sale, with the highest bidder winning the right to buy the product. Three approaches that were considered for the design of maintaining the anonymity of web transactions are described next.

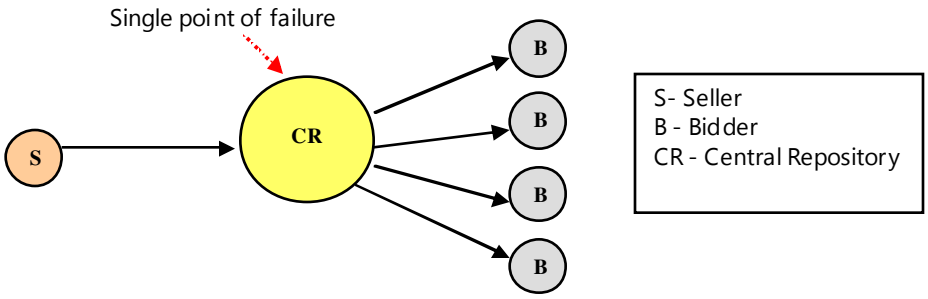


Figure 1. Dependency on a Central Repository, indicating a single point of failure.

### *2.3.1. Approach 1: Complete Dependency on a Central Repository*

Figure 1 illustrates this approach. Only one seller is shown for simplicity. In the case of multiple sellers, each of them must inform the central repository about the product they want to put up for sale.

Any club member can be a buyer or seller. The Central Repository (CR) maintains a database with the ratings of each buyer and seller based on her past transactions. If the buyer or seller violates certain rules of the club she has to pay a fine to the CR. The advantages of this approach are that the seller is unaware of the identity of the buyer and various buyers bid for the product with their anonymity maintained. The architecture is not complex; however, it does have some drawbacks.

First, there is a single point of failure, namely, the CR. The CR stores the club

members' ratings and controls solely all of the web transactions. Second, a large amount of bandwidth would be required for the web transactions to be performed efficiently.

### *2.3.2. Approach 2: Reduced Dependency on a Central Repository*

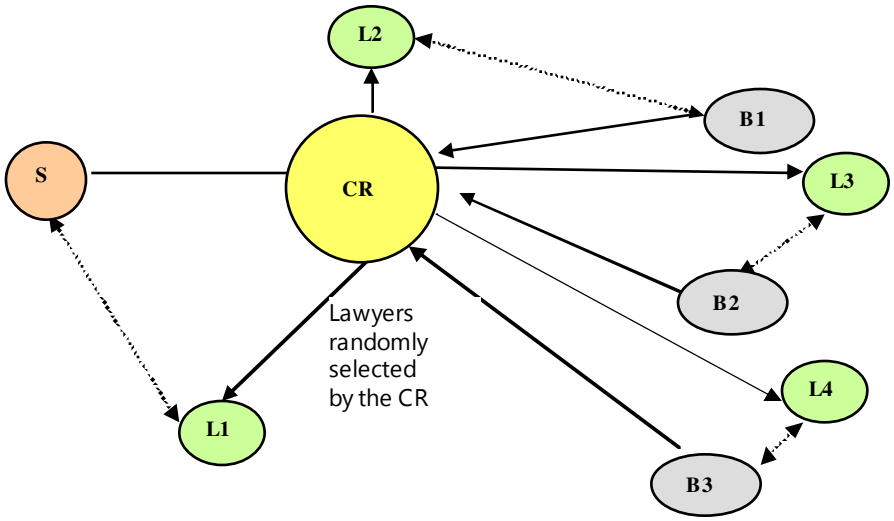


Figure 2. The CR is responsible for assigning a lawyer for each buyer.

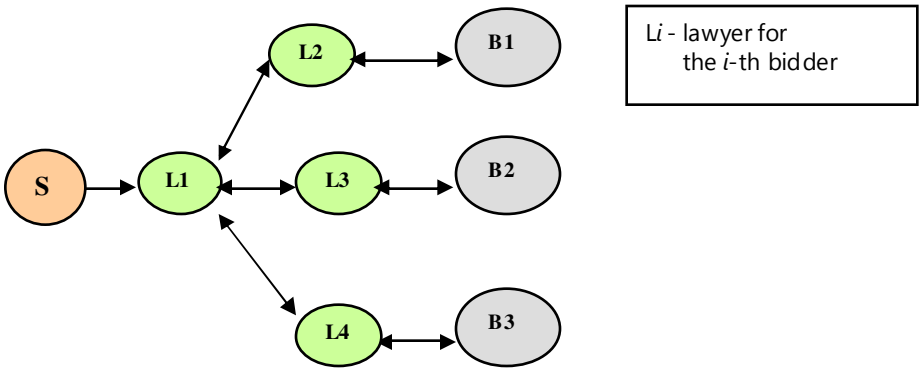


Figure 3. Once each buyer is assigned a lawyer, the CR no longer participates in the web transaction.

The term *lawyer* is used to denote a club member who himself does not wish to bid but who bids on behalf of other club members who want to maintain anonymity while bidding. Figure 2 illustrates an approach in which the CR assigns a lawyer to each bidder.

In this approach, the CR assigns a lawyer to each buyer and seller (see Figure 2). These lawyers then perform the transaction. In the example shown in Figure 3, the

lawyer L2 knows that B1 is a bidder, and the lawyer L1 knows that S is the seller. L2 does not have any other information about the other participants in the web transaction, i.e., the other bidders and the actual seller of the product. As shown in Figure 3, the CR no longer participates in the web transaction, and the anonymity of the seller and the bidders are still maintained.

This approach has its advantages. First of all, the dependency on the CR is largely reduced, and the amount of Internet resources is reduced compared to our previous approach. This is because, once the CR has assigned a lawyer to the seller and to each bidder, the CR no longer participates in the web transaction unless a fraud—such as a privacy violation by a lawyer, bidder, or seller—is reported. However, the number of buyers in the club for a particular product should be less than the number of lawyers. Note that the role of the CR cannot be ignored even though it is reduced, and that we still have a single point of failure.

### *2.3.3. Approach 3: Using Opportunistic Networks*

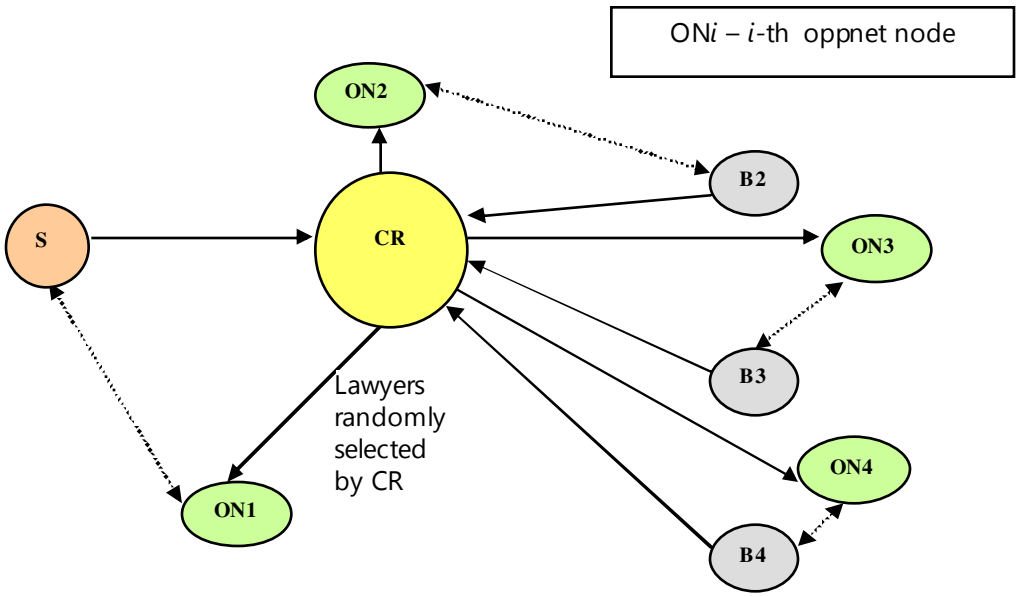


Figure 4. Using opportunistic networks as lawyers. In this approach, the number of lawyers in the club need not be more than the number of bidders.

In the above-mentioned approach, the number of bidders for a particular product must be less than the number of lawyers for the club mechanism to function. In this section, we propose a remedy to this limitation by using the approach used in opportunistic networks known as *helpers* [LiKG06, LKBG06]. In this approach (cf. Figure 4), the club members using the capabilities of *oppnets* may invite other nodes (who become helpers) that are not a part of the club to perform the function of a lawyer. The new node is assigned this lawyer role only and cannot participate as a bidder.

This approach ensures that the anonymity of the participating bidders is main-

tained. The helpers do not have any idea about the other participating nodes and must leave the club once their task is done.

### 2.4. Comparison of Discussed Design Methods

In the approaches discussed above, we rejected the *Complete Dependency on a CR* (Approach 1) and the *Reduced Dependency on a CR* (Approach 2) due to their dependency on a CR leading to a single point of failure. In Approach 2, even though the dependency is reduced, the number of lawyers must be greater than the number of participants in the web transaction for the club mechanism to function. By using opportunistic networks with helpers (Approach 3), we overcome these drawbacks.

## 3. Preserving Privacy in Web Transactions

### 3.1. Introduction

In this section, we discuss a solution to preserve the privacy of sensitive data based on a scheme for *privacy-preserving data dissemination* [LiBh06]. The *owner* of the data is an individual, a system, or an institution. The proposed scheme ensures that the data shared on the web is controlled by its owner. We use the term *guardian* to describe an entity that the owner trusts with the collection, processing, storage, and dissemination of the sensitive data. A guardian may pass sensitive data to a subsequent guardian. The risks of privacy violations grow when each guardian shares this private data.

Section 3.2 presents an overview of the Privacy-Preserving Data Dissemination (P2D2) approach. Section 3.3 describes semantic web rule languages, and Section 3.4 describes a few design approaches that we considered for maintaining the privacy of sensitive data in a web transaction.

### 3.2. Privacy-Preserving Data Dissemination

We first need to indicate that an entity can gain a higher level of trust in the eyes of another entity by sharing some of its sensitive information with the other entity. As an example, when a person downloads a trial version of software from a software distributing website, he needs to provide sensitive information such as email address, home address, phone number, city and zip code. The website then sends the key for the software the individual downloaded to his email address. In this example, an individual shares sensitive information with a website in order to gain its trust by establishing that he is a genuine user.

In a web transaction, one of the interacting parties is *stronger*; therefore the *weaker* party may need to share private information to gain a higher level of trust. The stronger party may choose to share this information with other parties, thus leaving the weaker party with little or no control over its private information. The idea proposed by this scheme is to *bundle,* or bind, this sensitive data with *metadata,* or rules, that must be followed by sharing this sensitive data. The metadata must be agreed upon by the owner of the sensitive data. The proposed scheme is of great importance for healthcare providers, researchers, online banking systems, and for customers and businesses who exchange sensitive information, like credit card numbers.

The terms "*sensitive data*" or "*private data*" may be used interchangeably.

We now discuss the operation of the proposed scheme in a healthcare environment [LiBh06]. If a patient or data owner provides her health information to a nurse via an *electronic health record* (EHR) application (primary guardian), then the EHR application obtains the patient's privacy preferences, and creates a bundle consisting of the patient's data coupled with the metadata, which includes the patient's preferences and hospital's policies. If the data are distributed further, they must include the entire bundle. Passing a bundle to a subsequent guardian is permitted only if it complies with the patient's privacy preferences and hospital policies.

If a bundle enters an environment where the conditions do not comply with those specified in the metadata, the contents must not be revealed. P2D2 proposes two approaches: *apoptosis* and *adaptive evaporation*. In the former, either all or no data are revealed. In the latter, part of the data is revealed while the rest is not. These two approaches are discussed further in Section 3.4.2. We require semantic web languages to define rules and other policies for data dissemination in the metadata. We must then parse these rules, and depending on the environmental conditions, we must be able to infer whether or not the data should be shared. A rule engine is required for this purpose.

## 3.3. Semantic Web Languages

The *Semantic Web* is a mesh of information linked up in such a way as to be easily processable by machines on a global scale. It can be thought of as being an efficient way of representing data on the World Wide Web or as a globally linked database [Palm01]. The Semantic Web is generally built on syntaxes that use Uniform Resource Identifiers (URIs) to represent data. A URI is a web identifier, like the strings starting with "http:" or "ftp:" [Palm01]. The Resource Description Framework (RDF) utilizes three URIs to represent data on the web. Once information is in the RDF form, it becomes easy to machine-process it, since RDF is a generic format and has many existing parsers.

When information needs to be processed (as opposed to situations in which the content only needs to be presented), we need a language to represent the meaning of terms in vocabularies and the relationships between those terms. This representation of terms and their interrelationships is called *ontology* [McFr04]. The OWL Web Ontology Language is used in situations in which data need to be processed automatically, such as in P2D2. In P2D2, we need to include some rules, defined by the owner of the data and the guardians. The Semantic Web Rule Language (SWRL) can be used for defining such rules. We also require a rule-based inference engine that can infer whether or not the data must be shared, given the metadata and current environmental conditions. We decided to use Jena as this engine. Jena is a Java framework for building Semantic Web applications.

## 3.4. Solutions for Anonymizing Participants in Web Transactions

Two approaches that were considered for maintaining the privacy of sensitive data in web transactions are described next. The terms *rules* and *conditions* are used interchangeably.

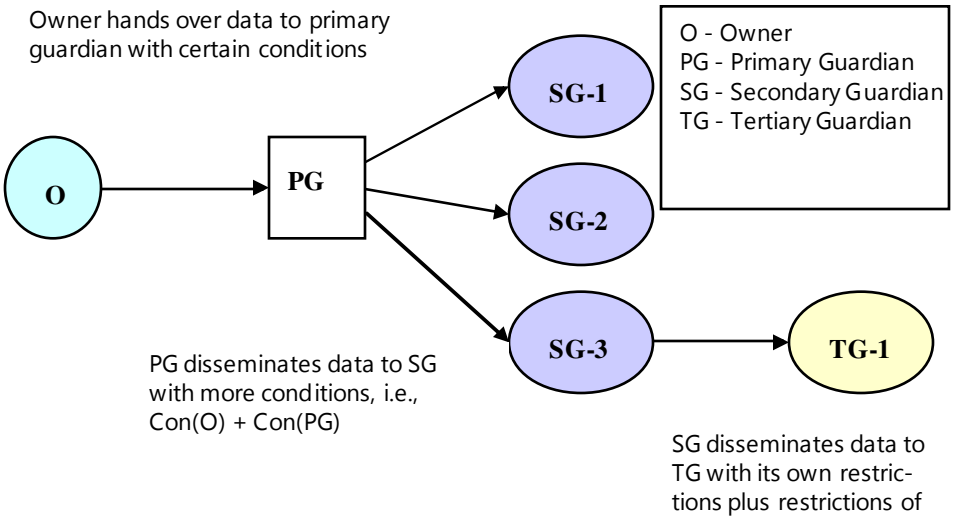


Figure 5. Sharing of data between guardians.

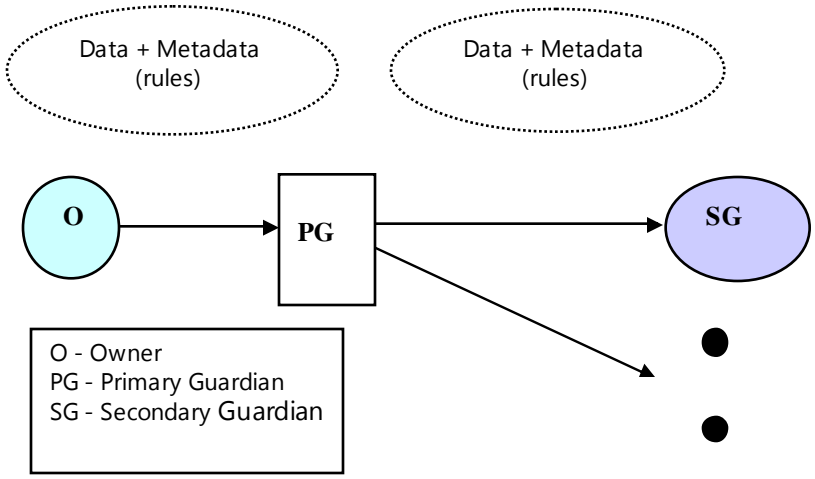


Figure 6. Dissemination of data and metadata among guardians.

### *3.4.1. Approach 1: No Bundling of Data and Metadata*

This section describes an approach in which the data and metadata are not bundled together. At each stage, the intermediate guardians must make sure that the data are shared conforming to the metadata. Figure 5 shows how the data are disseminated from the owner (O) to the primary, secondary, and tertiary guardians.

Characteristics of an approach in which data and metadata are *not* bundled together can be described as follows:

1. The owner specifies some conditions that must be met before the PG disseminates the owner's data to a secondary guardian.

2. It is then the responsibility of the PG to check whether the SG is a legitimate user or not.
3. The PG adds its own conditions before permitting data transfer, which, coupled with the conditions posed by the owner, must be met by the SG.
4. The cycle continues, with the SG then adding its own conditions, along with those of the O and the PG, before passing information to a TG.
5. Note that, before passing the data to the next guardian, at each interface certain conditions have to be checked.

A question that arises is: how do we trust these guardians? The approach discussed next bundles metadata and data so that we do not depend on a guardian for preserving privacy in data dissemination.

*3.4.2. Approach 2: Bundling Data and Metadata*

In this approach, we combine data and metadata into a single bundle. Figure 6 shows how this is done. An inference engine is used to decide whether the bundle can be shared or not. Metadata has to be coded in a Semantic Web standard so that it can be parsed by the inference engine.

Characteristics of the approach in which the data and metadata *are* bundled together can be described as follows:

1. In this approach, we do not need to depend on the intermediate guardians to check whether the conditions stated by the owner are met or not.
2. The PG, SG, etc., do not add their own conditions but simply transfer data when asked for them.
3. The data itself contain software that checks for certain conditions that must be checked before allowing the user to access it.
4. The data must be encrypted.

In the approaches discussed above, we specify ways in which the rules/conditions in the metadata must be verified before sharing the data. However, if these rules/ conditions are not met, either part or the entire data must not be disseminated. We discuss two mechanisms to prevent the dissemination of sensitive data in web transactions that do not satisfy the rules/conditions in the metadata. These are *apoptosis* and *adaptive evaporation*.

*A. Apoptosis*

A bundle about to be compromised chooses apoptosis over risking a privacy disclosure. In this approach, apoptosis destroys both data *and* metadata to prevent inferences from metadata. The *apoptosis mechanism* within a bundle can be implemented as a set of detectors setting off the associated apoptosis code. The code is activated when detectors determine a credible threat of a successful attack on the bundle by any host. Detectors find the bundle's trust level for a host based on information from multiple sources. These sources include a reputation databases, a source guardian's first-hand experience, and its second-hand opinions obtained from neighbors [LiBh06].

A detector in a bundle scheduled to arrive at a host with a trust level below a certain threshold will discover danger and will trigger apoptosis. There are different apoptosis threshold levels for hosts with different access permissions to private data.

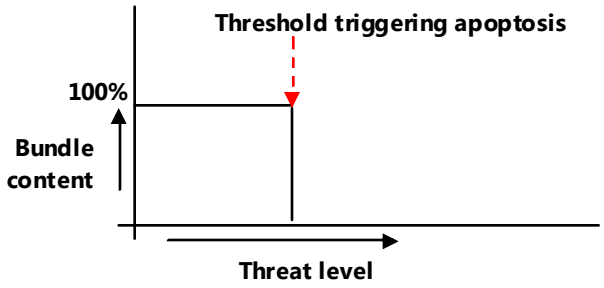


Figure 7. Apoptosis

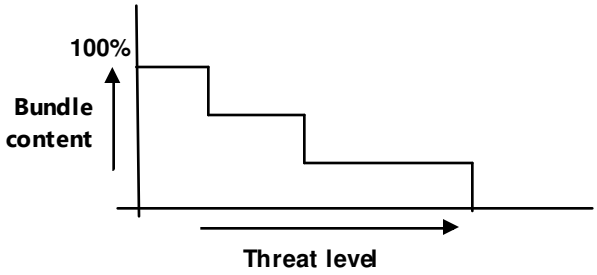


Figure 8. Adaptive evaporation.

For example, higher trust levels are usually expected of the patient's home clinic than from a clinic visited by a vacationing patient [LiBh06].

Figure 7 illustrates the mechanism of apoptosis in which the bundle self-destructs once the threshold limit for the threat level is reached. At the moment when apoptosis occurs, the bundle content goes from 100% to 0%.

### *B. Adaptive Evaporation*

Perfect passing of bundles is not always desirable. If bundles can be captured by attackers, their owners want to see their data evaporated partially (e.g., have them de-identified) with the most sensitive data evaporating first. The more threatening the environment, the larger the portion of the bundle that evaporates. To prevent inferences from metadata, all metadata evaporate in step with the associated data and in a manner that does not compromise data privacy in any other way. For instance, an owner's preferences for the owner's data never evaporate earlier than the data they protect [LiBh06]. Figure 8 illustrates this mechanism.

A number of different metrics were considered for the adaptive control of the evaporation. First, the trust level can be obtained—as discussed for apoptosis—and used to control the required degree of evaporation. Second, in some environments, trust is directly proportional to the *physical distance* from the data owner. Third, distance can be defined in a more sophisticated way, such as in terms of *data dissemination hops* [LiBh06].

Instances of data evaporation include replacing exact data with approximate data, or up-to-date values with outdated values. Evaporation can be applied to images as well. For example, a close-up photo of a person can be replaced with a distant whole-

body photo [LiBh06].

Apoptosis can be considered a special case of adaptive evaporation, which follows a step function with a constant minimum value (no evaporation) initially and the maximum value (complete evaporation) above a certain threshold.

# 4. Solution for Anonymizing Participants in Web Transactions

In this section we discuss existing systems providing anonymity to users. We analyze them for methods, algorithms, or protocols that will be useful for providing an optimal solution to maintain anonymity of the participants during a web transaction. We first discuss methods available in current research and publications before putting forward our proposed solution.

## 4.1. Analysis of Existing Methods

In this section we discuss existing systems that are useful for maintaining participant anonymity in web transactions.

### *4.1.1 Tor*

The primary goal of Tor [TorO06] is to prevent traffic analysis. This means that an attacker should not be able to trace the originator and the destination of a message if she intercepts a part of the message. The Internet data packet consists of a header that carries control information and the data payload. The data payload can be encrypted, but the header can reveal important information such as the data source, destination, and the size of the data. The identity of the sender can be revealed to an attacker through this packet header, hence forcing the sender to lose his anonymity. Encryption will not help prevent traffic analysis as we can only encrypt the data payload.

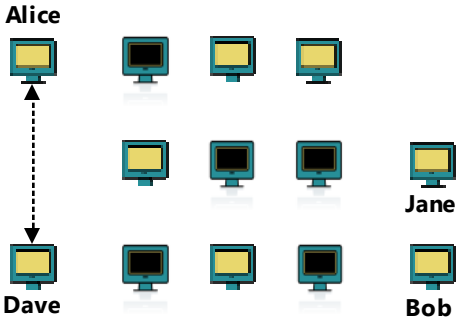


Figure 9a. Step 1 of Tor operation: Alice's Tor client obtains a list of Tor nodes from the directory server Dave (cf. (TorO06]). Tor nodes are shown as computers with black screens. The dotted-line arrow represents a non-encrypted link.

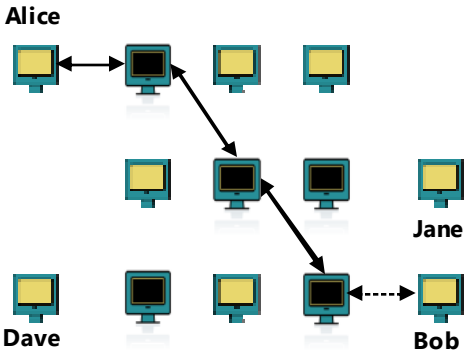


Figure 9b. Step 2 of Tor operation: Alice's Tor client picks a random path to the destination server Bob. Solid arrows represent encrypted links, and dotted-line arrows represent non-encrypted links (cf. [TorO06]).

The design concept used by Tor is similar to using a convoluted, hard-to-follow route in order to throw off a pursuer, and then periodically erasing one's footprints.

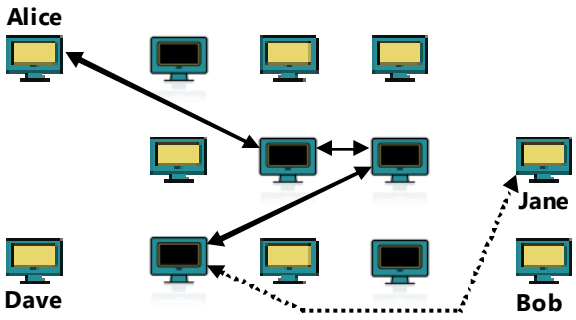


Figure 9c. Step 3 of Tor operation: If Alice wants to access Jane's site now, Alice's Tor client selects random path to Jane's site (cf. [TorO06]).

Instead of taking a direct route from source to the destination, the data packets on the Tor network take a random pathway through several servers that cover their tracks so no observer at any single point can tell where the data came from or where they are going.

To create a private network pathway with Tor, the user's software or client incrementally builds a circuit of encrypted connections through servers on the network. This is shown in Figure 9a, in which Alice's Tor client obtains a list of Tor nodes from the directory server Dave. Since the circuit is incremented one hop at a time, and each server along the way knows only the server that gave it data and the server to which it is giving data, no individual server ever knows the complete path that a data packet has taken. At each hop, the client uses a separate set of encryption keys to ensure that each hop cannot trace these connections as they pass through.

Once a circuit has been established, many kinds of data can be exchanged and several different sorts of software applications can be deployed over the Tor network. Since each server sees no more than a single hop in the circuit, it is impossible for an eavesdropper to use traffic analysis to link the connection's source and destination. Figure 9b shows how Alice's Tor client communicates with the server Bob. In Figure 9c, Alice's Tor client communicates with a different server, Jane, using a different path. Note that in Figures 9a through 9c the Tor nodes (shown as computers with black screens) in the private network selected by Alice's Tor client remain the same, though the path taken is different.

### *4.1.2. LPWA—Lucent Personalized Web Assistant*

The LPWA [LPWA00] is a tool that is used for personalized services on the web. Personalized web services are web pages that are tailored to the individual user. An example is a personalized news webpage, where a user specifies the type of news articles of interest to the user and the website displays only such articles. To achieve this, the website requires the user to create an account first so that it can store the preferences and associate them with a particular user. The drawback to this approach is that

these user accounts may be used by other parties as a means of deducing the user's browsing habits. Other user information (including the user's current location) can be made available to the websites due to the nature of the HTTP protocol and the cookie mechanism. Many websites also send junk email based on the browsing preferences of the users. Thus, there is a need to provide a service that can prevent users from being "recognized" when they return to their user accounts. Further, this will help in reducing junk email, a fast growing nuisance for web users.

The LPWA system consists of 3 parts: a Persona generator, a browsing proxy, and an email forwarder. The Persona generator consists of the *Janus* function designed to support pseudonyms. The LPWA email forwarder creates an alias address for a user when the user provides the @ escape sequence. As a part of the persona generator, a user obtains a different and seemingly unrelated alias email address for each website for which he is registered. For example, a user might be known as abcd007@lpwa.com at www.example.com and as gobroncos@lpwa.com at www.cnn.com. Whenever the email forwarder decrypts an alias email address in order to forward a message to a user's real email address, it includes the alias email address in the CC email header. If example.com is sold to spammers, the user can use a mail filter for the alias abcd007 and thus eliminate all emails received from these spammers; at the same time email messages from other sites are unaffected.

### *4.1.3. Crowds*

A web server can record information about users who visit it. These data include the IP address and thus the user's domain and workplace and her approximate location. Some web servers can link multiple sessions by the same user by planting a unique cookie in the user's browser. Thus, even if the user changes his/her location and visits the web site from different IP addresses, the web server can track the user's whereabouts. Most importantly, the same monitoring capabilities are made available to other parties as well (besides the web server). These include the user's Internet Service Provider (ISP) or the local gateway administrator who can observe all communication in which the user participates. Crowds [ReRu97] is a system that enables retrieval of information over the web without revealing any private data of the parties. The primary goal of Crowds is to increase the anonymity of the users on the web and make web browsing anonymous.

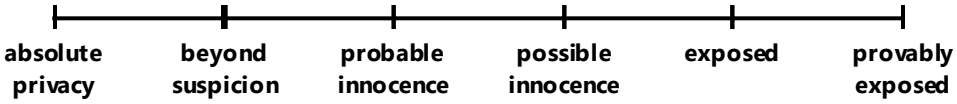


Figure 10. Various degrees of anonymity provided by Crowds. The degree of anonymity decreases as we move from left to right (cf. [ReRu97]).

The basic idea behind Crowds is to hide the actions of a user within the actions of many others. To execute a web transaction the user needs to join a crowd. The user's request to a web server is then passed to a random member in the crowd. That member can either commit or forward it to another randomly chosen member. When the

request is eventually submitted, it is submitted by a random member. Even the crowd members cannot identify the initiator of the request. There are several degrees of anonymity provided by crowds as shown in Figure 10. The degree of anonymity decreases as we move towards the right.

We need to explain a few degrees of anonymity named in Figure 10. The degree *beyond suspicion* indicates that the sender's (can be extended to receiver's) identity is beyond suspicion even if the attacker accesses the sent message. The chances that the sender is the originator of the message are no more than any other member in the crowd. The degree *probable innocence* indicates that the sender appears to be the likely originator of the message as much as he is unlikely to be the originator. Here, the attacker may have a reason to expect that the sender is the originator; however, it appears to him at least as likely that the sender is not the originator. The degree *possible innocence* indicates that a sender is possibly innocent if, from the attacker's point of view, there is a nontrivial probability that the originator is someone else. An advantage of Crowds is that each user actively participates in the function of the crowd, hence increasing the throughput. Also, if a new member joins the crowd, the load on each user's computer in the crowd remains roughly constant. However, in the mix, the load of each server increases proportionally with the number of users, hence decreasing the throughput.

A user is represented in the crowd by a process called a *jondo* (the term is derived from "John Doe," a synonym for an anonymous person). The user then contacts the server, called a *blender*, to request admittance into the crowd. The blender then reports to this jondo the current membership of the crowd and the information that enables the jondo to participate in the crowd. The function of the blender is to put a requesting user into the crowd, and is not needed later. The request is issued by the browser, forwarded through a number of jondos, and eventually submitted to the end server. The sequence of jondos that a request traverses is called a *path*. An important feature of the Crowds protocol is that the request is sent in the same form along each "hop" of the path, so that each jondo cannot tell whether its predecessor initiated the request or is just forwarding it from another jondo. The server's reply to the request is usually a web page which is sent backwards through the same path. Subsequent requests initiated by the same jondo follow the same path through the crowd, even if these requests are targeted for different web servers. That is, once established, a path remains static as long as possible unless a jondo on a path fails or new jondos are added. In these cases, the paths of all jondos are forgotten and rerouted from scratch.

#### *4.1.4. Tarzan*

Tarzan [FrMo02] is a peer-to-peer anonymous IP network overlay. It achieves its anonymity with layered encryption and multi-hop routing. A message initiator chooses a path of peers randomly through a restricted topology in a way that adversaries cannot easily influence. Its goal is to allow a host to communicate with an arbitrary server in such a manner that nobody can determine the host's identity.

Consider a host H that sends a message to a server through a proxy, such as Anonymizer.com. This system fails if the proxy reveals a user's identity or if an adversary can observe the proxy's traffic. Typically Tarzan works as a three step process. First, a node running an application that desires anonymity selects a set of nodes to for a

path through the overlay network. Next, this source-routing node establishes a tunnel using these nodes, which includes distribution of session keys. Finally, it routes data packets through this tunnel. The exit point of the tunnel is a Network Address Translator (NAT). This NAT forwards the anonymized packets to servers that are not aware of Tarzan, and it receives a response from the servers and reroutes the packets via this tunnel.

A Tarzan tunnel passes two distinct types of messages between nodes: data packets and control packets. A flow tag uniquely identifies each link of each tunnel. A relay rapidly determines how to route a packet tag.

A tunnel setup in Tarzan is done as follows. The sender pseudo-randomly selects a series of nodes from the network based on its local topology. The tunnel fails if one of its relays stops forwarding packets. To overcome this, the initiator sends ping messages to the PNAT (server-side Pseudonymous Network Address Translator) through the tunnel and waits for acknowledgements. The initiator will then have to determine the point of failure, if it does not receive a response. Tarzan uses a simple gossip-based protocol for *peer discovery*. A node can prune inactive neighbors when they do not respond to cover traffic establishment requests. Once peers are discovered, *peer selection* uses a three level hierarchy: first among all */16* subnets, then among */24* subnets belonging to this 16-bit IP address, then among the relevant IP addresses. (The */n subnet* of a network is the subnet with addresses determined by the last *n* bits of the mask.) The originator node will then request the selected peer to exchange bi-directional mimic data with it.

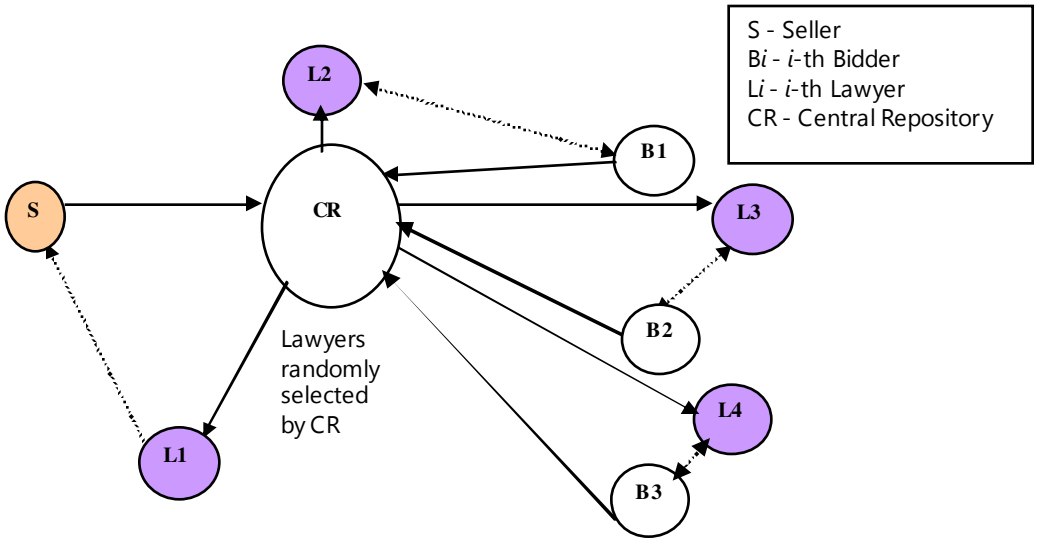


Figure 11a. Club mechanism with economic incentives.

## 4.2. Proposed Design for Maintaining Anonymity of Participants in Web Transactions

In this section we propose a method to maintain the anonymity of participants in web transactions using a modified form of the *club mechanism with economic incen-*

*tives* [JeLB04], as shown in Figure 11a.

The CR is comprised of a group of elite agents known as *super agents* as shown in Figure 11b. The entire club is subdivided into sections and each super agent is assigned one sub section. A super agent is a club member who can be nominated based on factors such as availability and internet connection speed. If one super agent fails, only the agents under it would not be able to participate in the web transactions taking place within the club, whereas all other agents not under it could still participate. The super agents themselves will decide on the initiation fee and the fines for violating agents. Note that one super agent need not inform the other super agents about the random agents chosen for the web transactions.

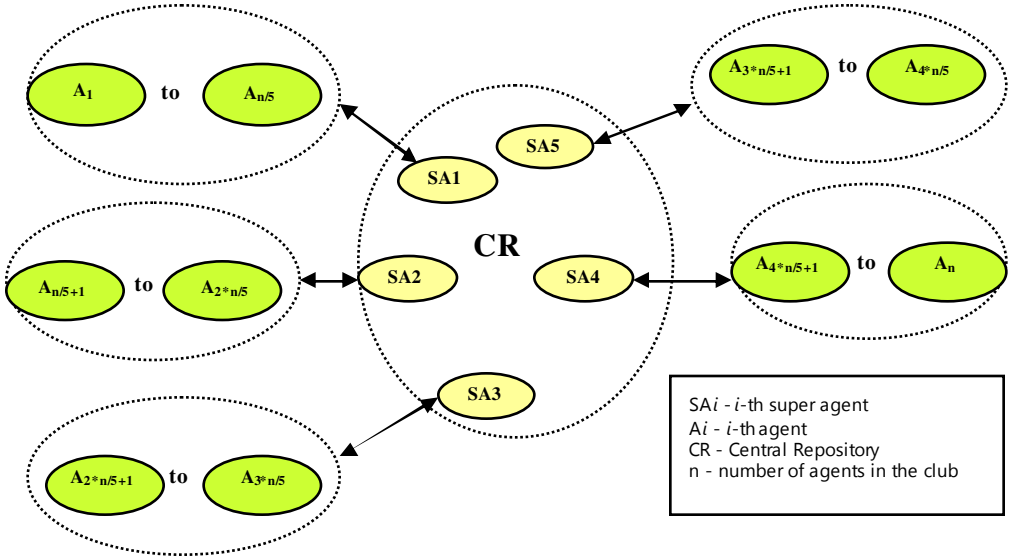


Figure 11b. Each CR (comprised of super agents) is responsible for the actions of all agents under it.

In Figure 11b, we have considered the CR comprised of five super agents. This is just an example, and the number of super agents may vary depending on factors such as the number of agents in the club. The super agents can invite not only club members but, in special cases, also non-members. The special cases arise when the number of club agents participating in a web transaction exceeds the number of those who do not. In this case, the super agents can invite nodes from outside the club to perform the roles of anonymizing agents or lawyers. These opportunistic nodes (or helpers) are invited only for a particular task, after which they must leave the club. The super nodes may even decide to use more than two lawyers to carry out a web transaction, further increasing the anonymity of the sender and buyer.

## 4.3. Comparison of Proposed and Analyzed Methods

Table 2 below provides a comparison of the club mechanism with economic incentives, Tor, Crowds and Tarzan. Tor and Tarzan have a single point of failure, that is, if the primary server goes down, the entire system fails and hence these are dependent entirely on the primary server. In the club mechanism with economic incentives, if one of the super agent servers goes down, only the agents that were depend-

ent on this super agent will suffer, leaving the other agents active. Load balancing is thus not possible in Tor and Tarzan due to the nature of their architecture, i.e., dependency on a single server.

Anonymity of senders and receivers is one of the goals of the club mechanism with economic incentives. Crowds, Tor and Tarzan may provide sender anonymity whereas Crowds also provides receiver anonymity. Payload encryption may be done in the club mechanism with economic incentives if desired. When packets are to be forwarded, the intermediate nodes may decide to send the packets or not. In the club mechanism, the agents do have this option, with economic incentives at stake if they do forward them. The Crowds mechanism has an option in which jondos may choose not to forward packets if they wish. Finally, in the club mechanism with economic incentives, the route that a message takes is decided by the super agents. In the cases of Tor and Tarzan, the route is determined by the sender. In Crowds, the route taken by the message is performed dynamically.

Table 2. Comparison of technologies discussed in the design for maintaining anonymity of the participants

| | **Club mechanism with economic incentives** | **Tor** | **Crowds** | **Tarzan** | **LPWA** |
|---|---|---|---|---|---|
| **Single point of failure** | No | Yes - directory-based approach | No | Yes | NA |
| **Memory usage** | Low | High | Low | High | NA |
| **Switching** | Packet | Circuit | Packet | Circuit | NA |
| **Load balancing** | Possible | Not possible - dedicated tunnel set up | Possible | Not possible | NA |
| **Scaling complexity** | O(1) | O(n) | O(1) | O(n) | NA |
| **Sender anonymity** | Yes | Maybe | Maybe | Maybe | Yes |
| **Receiver anonymity** | Yes | No | Yes | No | Yes |
| **Payload encryption of packet from sender** | Maybe | No | Maybe | No | No |
| **Payload encryption in receiver's response** | Maybe | No | Maybe | No | No |
| **Intermediate nodes decide on packet forwarding?** | Yes | No | Yes | No | No |
| **Route selection** | By controller (as a set of super agents) | By sender | Dynamic | By sender | NA |

# 5. Solutions for Privacy--Preserving Dissemination of Sensitive Data in Web Transactions

In this section, we discuss a solution to preserve the privacy of sensitive data in web transactions. In a web transaction one of the involved participants is *stronger* than the other, therefore participants must agree on a set of rules or policies in order to interact successfully. This means that they need to negotiate before starting a web transaction [Ande04].

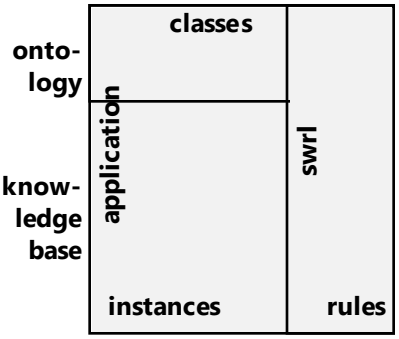


Figure 12a. A combination of rules, ontology classes and the data to be shared (cf. [Ande04]).

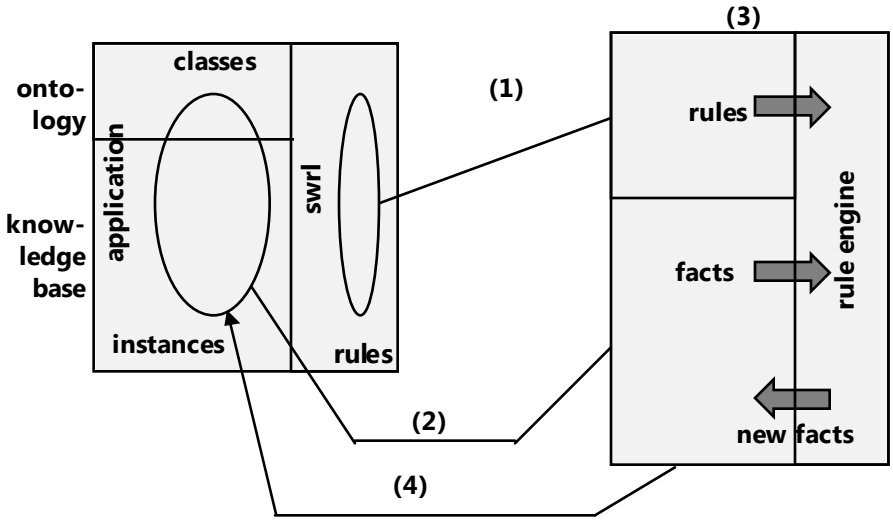


Figure 12b. Rules and facts are fed to an inference engine and inferences are made based on this (cf. [Ande04]).

The data and metadata (consisting of rules) are bundled and treated as an atomic unit. A web rule language is used to define the rules that must be adhered to while

sharing data in such web transactions. The Semantic Web Rule Language (SWRL) is one such option and we describe it briefly.

## 5.1. SWRL– Semantic Web Rule Language

In this section we will describe SWRL [HPBT04], a web standard used to define rules on the web. SWRL enables to combine Horn-like rules (in the form of an implication, as shown below) with an OWL knowledge base. The rules in SWRL are stated as follows: axiom::= rule. A human readable syntax of the rules is shown as an implication: *head* => *body*. Rules with an empty *head* (or *antecedent*) and non-empty *body* (or *consequent*) are used to provide unconditional facts. The head and the body may have zero or more atoms (indivisible elements). *Atoms* can be of the forms *C(x)*, *P(x, y)*, *sameAs(x, y)*, etc., where *C* is an OWL description, *P* is an OWL property, and *x* and *y* are either variables, OWL individuals or OWL data values. The non-terminals are shown in bold and not quoted.

In our solution, rules are specified by the owner of the data. If these rules are not satisfied, the data should be disseminated only partially (e.g., without the most sensitive data) or not disseminated at all. SWRL requires a rule engine for its execution. Based on the rules and the current data, the work of the rule engine is to infer whether or not the data must be disseminated. A structure of a rule engine is shown in Figures 12a and 12b.

Figure 12a shows a combination of rules, instances and classes. The rules are those defined by the owner of the sensitive data and are included in the metadata. The instances refer to the current conditions, and the classes belong to the ontology being used. Figure 12b shows how an inference engine is used to deduce new facts, given the current facts and rules.

Jena is one rule engine that supports SWRL. It is a Java-based framework. Based on the inferences made by the rule engine, a decision as to whether or not the data is shared further must be made; if any data is shared, a decision how much of the data must be hidden is needed as well.

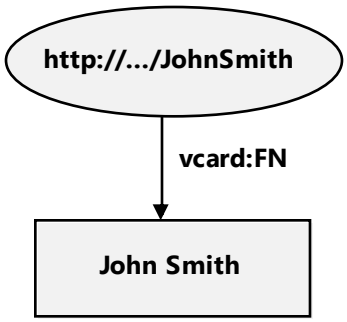


Figure 13. An example illustrating a Jena resource (cf. [McBr05]).

## 5.2. Jena – Rule Engine Supporting SWRL and Other Web-Rule Languages

Jena is a Java framework for building semantic web applications [Reyn06]. RDF (Resource Description Framework) is a web standard for describing resources [McCa04]. A resource is simply something that can be identified. For example, a university can be identified by its address, telephone number, etc. The Resource Description Framework has now become a W3C recommendation, joining other important Web standards such as XML and SOAP [McCa04]. Consider the example in which John Smith is identified by his visiting card, a VCard. Figure 13 represents this diagrammatically.

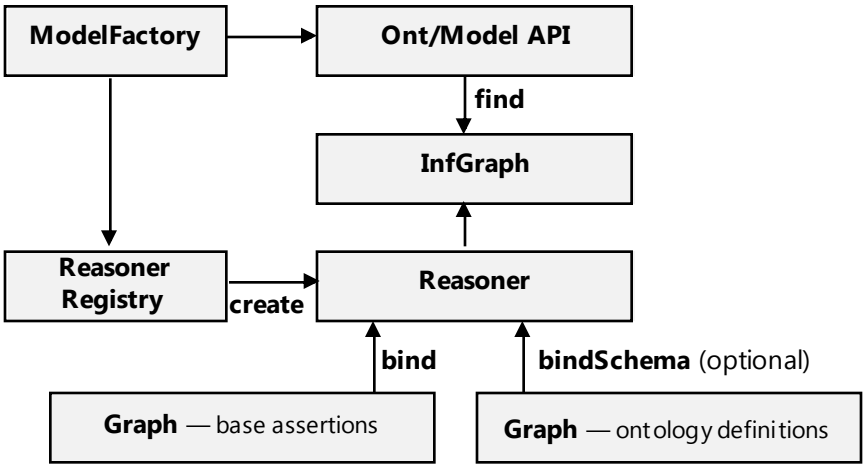


Figure 14. Inference support in Jena (cf. [Reyn06]).

The resource is John Smith and he is identified by a URI (Uniform Resource Identifier). Resources have properties, one of which is his full name on the VCard. The part before the ":" is called the *namespace prefix* and represents a namespace. The part after the ":" is called a *local name* and represents a name in that namespace. Each property has a value. In this case the value is a literal.

### *5.2.1. Inference Support in Jena*

In this section, we describe the inference support provided by Jena. Figure 14 describes the architecture of the Inference Model used by Jena [Reyn06].

First, we need to create a model consisting of the rules, the data and ontology. There are five different reasoners in Java that can be used. One of them has to be selected from the reasoner registry, as shown in Figure 14. Once the reasoner has been selected, we need to provide it with the ontology definitions/rules and the data/facts. The reasoner then creates an inference graph based on this input. Second, a Java API is used to query for information from this inference graph. Third, based on the results from the inference graph, we can then decide whether or not the data is to be shared.

We now provide a simple example (adapted from [Reyn06]) of Java code showing

rules written in Jena, and the inferences resulting from the rules. Suppose that property "p" is a sub-property of another property "q" and there is a resource "a" with a value "hi" for "p". Comments are bracketed by /* and */. The rest is Java code.

Jena code:

```
String NS = "urn:x-hp-jena:eg/";
        /* Build a trivial example data set */
 Model rdfsExample = ModelFactory.createDefaultModel();
        /* Initialize two properties: p and q */
 Property p = rdfsExample.createProperty(NS, "p");
 Property q = rdfsExample.createProperty(NS, "q");
        /* Add the rule: p is a sub property of q */
 rdfsExample.add(p, RDFS.subPropertyOf, q);
        /* Create a resource A with the property p having the value hi */
 rdfsExample.createResource(NS+"a").addProperty(p, "hi");
        /* Create an inference model */
InfModel inf = ModelFactory.createRDFSModel(rdfsExample);
        /* The resulting model shows that "a" also has property "q" of
        value "hi" by virtue of subPropertyOf entailment. */
Resource a = inf.getResource(NS+"a");
System.out.println("Statement: " + a.getProperty(q));
```

Output showing that the resource "a" has a property "q" using the sub-property entailment:

*Statement: [urn:x-hp-jena:eg/a, urn:x-hp-jena:eg/q, Literal]*

The above code is trivial and by no means reflects the power of the inference engine. However, it does give an idea about the basic steps that lead to creating an inference model and querying it.

## 6. Conclusion

We have discussed schemes to maintain anonymity and privacy of participants in web transactions. We have proposed a modified form of the *club mechanism with economic incentives*. The concept of having super agents as part of the Central Repository makes the existing club mechanism far more distributed. We have also proposed the use of opportunistic networks in playing the role of lawyers (members' agents) in the club mechanism. In this way, the number of club members participating in one particular web transaction can exceed the number of potential lawyers (i.e., club members who do not need to participate in that web transaction). Further, there is no single point of failure; if a super agent fails, only the agents in the network controlled by it will be unable to participate in web transactions, whereas the rest of the network functions normally.

We have compared existing systems that provide participant anonymity in web transactions to the modified club mechanism with economic incentives. We discussed the advantages and drawbacks of the systems and have presented reasons as to why

this club mechanism is the best choice. The lack of understanding of incentives for encouraging group cooperation is a major drawback in systems other than the club mechanism with economic incentives [JeLB04].

We discussed *privacy-preserving data dissemination* (P2D2) to maintain privacy of data in a web transaction. Typically, in any web transaction, one of the participants is always stronger than the other (in terms of the power to ask the other party for sensitive data). Dissemination of sensitive data owned by the weaker partner must be controlled. The proposed scheme for privacy-preserving data dissemination enables control of data by their owner [LiBh06]. This is accomplished by combining data and metadata into a bundle and then disseminating the entire bundle.

We discussed the use of semantic web languages to represent metadata. Metadata include a set of rules defined by the owner of the data. An inference engine is needed to control bundles as dictated by the rules within metadata. Based on the facts, e.g., the transaction participant identity to whom the data is being shared and the metadata, the inference engine deduces whether or not any data in the bundle can be disclosed. If some data can be disclosed, the inference engine must decide how much of the data can be disclosed.

We presented an overview of an inference engine named Jena, which is a Java framework for building and querying inference models in semantic web applications. We described a simple inference made by Jena. We feel that Jena will provide a strong option for implementing the P2D2 mechanism in the future.

## Acknowledgements

The authors are very grateful to the anonymous reviewers for their excellent corrections and suggestions. We also thank all editors, including Lotfi Ben Othmane, Ilse Schweitzer and Chris Triezenberg, for their professional help. All remaining mistakes or omissions are, of course, ours.

## References

[AbDH04] K. Aberer, A. Datta, and M. Hauswirth, "A decentralized public key infrastructure for customer-to-customer e-commerce," *J. Business Process Integrations and Management*, Volume X, No. X, 2004 (pre-publication version). Available at: http://lsirpeople.epfl.ch/aberer/PAPERS/IJBPIM2004.pdf

[AbDH05] K. Aberer, A. Datta, and M. Hauswirth, "A decentralized public key infrastructure for customer-to-customer e-commerce," *Int.J. Business Process Integrations and Management*, Vol. 1(1), 2005, pp. 26-33.

[ADDC05] C. Ardagna, E. Damiani, S. de Capitani di Vimercati, C. Fugazza, and P. Samarati, "Offline Expansion of XACML Policies," *Privacy and Identity Management for Europe* (*PRIME*) *Framework 6 Projects* (*FP6*), *Project reference Nr.:* IST-2002-507591, July 2005. Available at: http://seclab.dti.unimi.it/Papers/icswe05.pdf.

[Ande04] A. Anderson, "IEEE Policy 2004 - Workshop, Comparing WSPL and WS-Policy," Sun Labs, 2004, Burlington, MA.

[CDDF05] P. Ceravolo, E. Damiani, S. de Capitani di Vimercati, C. Fugazza, and P.

Samarati, "Advanced Metadata for Privacy-Aware Representation of Credentials," *Proc. 21st International Conference on Data Engineering Workshops* (*ICDEW'05*), 2005.

[CDEK04] C. Clifton, A. Doan, A. Elmagarmid, M. Kantarcioglu, G. Schadow, D. Suciu, and J. Vaidya, "Privacy-Preserving Data Integration and Sharing," *Proc. 9th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, 2004, pp. 19-26.

[DaVS05] E. Damiani, S. de Capitani di Vimercati, and P. Samarati, "New Paradigms for Access Control in Open Environments," *Privacy and Identity Management for Europe* (*PRIME*) *Framework 6 Projects* (*FP6*), *Project Reference Nr.:* IST-2002-507591, 2005. Available at: http://seclab.dti.unimi.it/Papers/isspit05.pdf

[Dura03] R. Duraikannu, "Abstract of Anonymity in Web Transactions," 2003. Available at: http://gaia.ecs.csus.edu/~ghansahi/classes/projects/502/duraikannu/Ramkumar_Duraikannu_finalreport_11_21_03.doc

[FrMo02] M.J. Freedman and R. Morris, "Tarzan: A Peer-to-Peer Anonymizing Network Layer," *Proc. 9th ACM Conference on Computer and Communications Security* (*CCS*), 2002, pp.193-206. Available at: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.19.9397

[GGKM99] E. Gabber, P. Gibbons, D. Kristol, Y. Mattias, and A. Mayer, "Consistent yet anonymous, web access with LPWA," *Communications of the ACM*, Vol. 42 (2), 1999, pp. 42-47.

[GPBS05] F.J. García Clemente, G. Martínez Pérez, J.A. Botía Blaya, and A.F. Gómez Skarmeta, "Representing Security Policies in Web Information Systems," *2nd Annual Workshop on the Weblogging Ecosystem: Aggregation, Analysis and Dynamics* (*WWW 2005*), Chiba, Japan, May 2005. Available at: http://www.csee.umbc.edu/pm4w/papers/clemente5.pdf

[HPBT04] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean, "SWRL: A Semantic Web Rule Language Combining OWL and RuleML," DAML report, May 2004. Available at: http://www.daml.org/2004/04/swrl/

[JeLB04] M. Jenamani, L. Lilien, and B. Bhargava, "Anonymizing Web services through a Club Mechanism with Economic Incentives," *Proc. IEEE International Conference on Web Services* (*ICWS'04*), San Diego, California, June 2004. Available at: http://csdl2.computer.org/persagen/DLAbsToc.jsp?resourcePath=/dl/proceedings/&toc=comp/proceedings/icws/2004/2167/00/2167toc.xml&DOI=10.1109/ICWS.2004.1314823

[KaSC04] A. Kapadia, G. Sampermane, and R.H. Campbell, "*Know* Why Your Access Was Denied: Regulating Feedback for Usable Security," *Proc. 11th ACM Conference on Computer and Communications Security*, 2004, pp. 52-61. Available at: http://www.cs.dartmouth.edu/~akapadia/ papers/know.pdf

[KDTC04] A. Abou El Kalam, Y. Deswarte, G. Trouessin, and E. Cordonnier, "A Generic approach for Healthcare Data Anonymization," *Proc. 2004 ACM Workshop on Privacy in the Electronic Society,* Washington, DC, 2004, pp. 31-32.

[LiBh06] L. Lilien and B. Bhargava ,"A Scheme for Privacy-preserving Data Dissemination," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 36(6), 2006, pp. 503-506. Available at http://ieeexplore.ieee.org/

iel5/3468/34230/01632285.pdf

[LiKG06] L. Lilien, Z.H. Kamal and A. Gupta, "Opportunistic Networks: Research Challenges in Specializing the P2P Paradigm," *Proc. 3rd International Workshop on P2P Data Management, Security and Trust* (*PDMST'06*), Kraków, Poland, Sep. 2006, pp. 722 –726.

[Linn05] J. Linn, "Technology and Web User Data Privacy," *IEEE Security & Privacy*, Vol. 3(1), Jan.-Feb. 2005, pp.52-58. Available at: http://ieeexplore.ieee.org/iel5/8013/30310/01392701.pdf?isnumber=&arnumber=1392701

[LKBG06] L. Lilien, Z.H. Kamal, V. Bhuse and A. Gupta, "Opportunistic Networks: The Concept and Research Challenges in Privacy and Security," *Proc. International Workshop on Research Challenges in Security and Privacy for Mobile and Wireless Networks* (*WSPWN 2006*), Miami, Florida, Mar. 2006, pp. 134-147.

[LPWA00] "The Lucent Personalized Web Assistant," A Bell Labs Technology Presentation. Last Modified on July 2000. Available at: http://www.bell-labs.com/project/lpwa/proxy_index.html.

[McBr05] B. McBride, "An Introduction to RDF and the Jena RDF API," 2005. Available at: http://jena.sourceforge.net/tutorial/RDF_API/

[McCa04] P. McCarthy, "Introduction to Jena," 2004. Available at: http://www.128.ibm.com/developerworks/java/library/j-jena/

[McFr04] D.L. McGuinness, F. van Harmelen, "OWL Web Ontology Language Overview," 2004. Available at: http://www.w3.org/TR/owl-features/

[Palm01] S. B. Palmer, "The Semantic Web: An Introduction," 2001. Available at: http://infomesh.net/2001/swintro/

[ReBE03] A. Rezgui, A. Bouguettaya, M.Y. Eltoweissy, "Privacy on the Web: Facts, Challenges, and Solutions," *IEEE Security and Privacy*, Vol. 1(6), Nov.-Dec. 2003, pp. 40-49. Available at: http://ieeexplore.ieee.org/iel5/8013/28051/01253567.pdf

[ReRu97] M.K. Reiter, A.D. Rubin, "Crowds: Anonymity for Web Transactions," *DIMACS Technical Report*, Center for Discrete Mathematics & Theoretical Computer Science, Rutgers, New Jersey, 1997. Available at: http://avirubin.com/crowds.pdf

[Reyn06] D. Reynolds, "Jena 2 Inference Support," 2006. Available at: http://jena.sourceforge.net/inference.

[Swee97] L. Sweeney, "Guaranteeing Anonymity when Sharing Medical Data, the Datafly System," 1997. Available at: http://www.amia.org/pubs/symposia/D004462.PDF

[TorO06] "Overview of Tor," 2006. Available at: http://tor.eff.org/overview.html.en

[VBFP04] V.S. Verykios, E. Bertino, I. Nai Fovino, L. Parasiliti Provenza, Y. Saygin, and Y. Theodoridis, "State-of-the-art in Privacy Preserving Data Mining," March 2004. Available at: http://www.unipi.gr/faculty/ytheod/pubs/journals/sigrec03.pdf