Private Data Protection in Ubiquitous Computing

Malika Yaici
Laboratoire LTII
University of Bejaia
Bejaia, 06000, Algeria
yaici_m@hotmail.com

Samia Ameza*, Ryma Houari† and Sabrina Hammachi‡
Computer Department, University of Bejaia
Bejaia, 06000, Algeria
*ameza_samia@yahoo.fr †ri.houari@hotmail.fr
‡hassiba_rima@yahoo.fr

***Abstract*—A system in ubiquitous computing consists of a large amount of heterogeneous users and devices that communicate with each other. Users in this dynamic field communicate with lightweight and autonomous devices, which accentuate security problems and make them more complex. The existing mechanisms and solutions are inadequate to address new challenges mainly for problems of authentication and protection of privacy. In this paper, a new security architecture called Tree Based distributed Privacy Protection System is proposed. It supports protection of users private data and addresses the shortcomings of systems like GAIA, OpenID and User-directed Privacy Protection (UPP). Furthermore, it takes into account the domain dissociation property, in order to achieve decentralized data protection.**

Keywords*–*Ubiquitous Computing; Security; Private Data Protection; Privacy; Integrity.

I. Introduction

The growing number of Internet users and the integration of mobile clients has changed distributed computer science, by allowing the creation of smart and communicating environments, thus offering to the user the opportunity to make interactions with its environment and its equipments easily and transparently leading to the concept of ubiquitous computing.

Its origins date back to 1991, when Mark Weiser [1] presented his futuristic vision of the 21st century computing by establishing the foundations of pervasive computing. It aims to integrate computer technology in man's everyday life in various fields (Health, Public services, etc.). To improve interactivity, it offers the user the ability to access various features and services of its environment and from any mobile device (personal digital assistant PDA, tablet computer, smartphone, etc). The emergence of these devices has created new security problems for which solutions and existing mechanisms are inadequate, especially concerning the problems of authentication and users' private data protection. In such a system, the existence of a centralized and homogeneous security policy is in fact not desirable. It is therefore necessary to give more autonomy to security systems, mainly by providing them with mechanisms establishing dynamic and flexible cooperation and collaboration.

Mobile devices and the Internet of Things (IoT) present some problems such as incorrect location information, privacy violation, and difficulty of end-user control. A conceptual model is presented in [2] to satisfy requirements which include a privacy-preserving location supporting protocol using wireless sensor networks for privacy-preserving child-care and safety where the end-user has authorized credentials anonymity.

In [3], the author uses the framework of contextual integrity related to privacy, developed by Nissenbaum in 2010 [4], as a tool to understand citizen's response to the implementation of IoT related technology in a supermarket. The purpose was to identify and understand specific changes in information practices brought about by the IoT that may be perceived as privacy violations. Issues identified included the mining of medical data, invasive targeted advertising, and loss of autonomy through marketing profiles or personal affect monitoring.

Dhasarathan et al. [5] present an intelligent model to protect user's valuable personal data based on multi-agents. A hybrid hash-based authentication technique as an end point lock is proposed. It is a composite model coupled with an anomaly detection interface algorithm for cloud user's privacy preserving (intrusion detection, unexpected activities in normal behavior).

In [6], the authors focus on information privacy protection in a post-release phase. Without entirely depending on the information collector, an information owner is provided with powerful means to control and audit how his/her released information will be used, by whom, and when. A set of innovative owner-controlled privacy protection and violation detection techniques have been proposed: Self-destroying File, Mutation Engine System, Automatic Receipt Collection, and Honey Token-based Privacy Violation Detection. A next generation privacy-enhanced operating system, which supports the proposed mechanisms, is introduced. Such a privacy-enhanced operating system stands for a technical breakthrough, which offers new features to existing operating systems.

Efficiency and scalability become critical criteria for privacy preserving protocols in the age of Big Data. In [7], a new Private Set Intersection (PSI) protocol, based on a novel approach called oblivious Bloom intersection is presented. The PSI problem consists of two parties, a client and a server, which want to jointly compute the intersection of their private input sets in a manner that at the end the client learns the intersection and the server learns nothing. The proposed protocol uses a two-party computation approach, which makes use of a new variant of Bloom filters called by the author Garbled Bloom filters, and the new approach is referred to as Oblivious Bloom Intersection.

Private Information Retrieval (PIR) protocols allow users to learn data items stored on a server which is not fully trusted, without disclosing to the server the particular data element retrieved. In [8], the author investigates the amount of data disclosed by the the most prominent PIR protocols during a single run. From this investigation, mechanisms that limit the PIR disclosure to a single data item are devised.

Releasing sensitive data while preserving privacy is a problem that has attracted considerable attention in recent years. One existing solution for addressing the problem is differential privacy, which requires that the data released reveals little information about whether any particular individual is present or absent from the data. To fulfill such a requirement, a typical approach adopted by the existing solutions is to publish a noisy version of the data instead of the original one. The author of [9] considers a fundamental problem that is frequently encountered in differentially private data publishing: Given a set D of tuples defined over a domain Ω, the aim is to decompose Ω into a set S of sub-domains and publish a noisy count of the tuples contained in each sub-domain, such that S and the noisy counts approximate the tuple distribution in D as accurately as possible. To remedy the deficiency of existing solutions, the author presents PrivTree, a histogram construction algorithm that adopts hierarchical decomposition but completely eliminates the dependency on a predefined limit h on the recursion depth in the splitting of Ω.

Middleware is an essential layer in the architecture of ubiquitous systems, and recently, more emphasis has been put on security middleware as an enabling component for ubiquitous applications. This is due to the high levels of personal and private data sharing in these systems. In [10], some representative security middleware approaches are reviewed and their various properties, characteristics, and challenges are highlighted.

The objective of our work is to develop an architecture that meets the security constraints of the ubiquitous systems that support the protection of user's private data. The idea is to consider the separation of different user data on separate domains, so that an intruder never reaches all of the user's private information and protect them against unauthorized and unwanted access and limit the transmission of such sensitive data.

The paper is organized as follows: after this introduction, some existing research works on the domain are presented in Section 2 (Ubiquitous environment security requirements) and Section 3 (GAIA, OpenID and UPP) with a comparison between them. Then, in Section 4, the proposed system is given with an illustrative example. An improved solution based on a tree structure is presented in Section 5 with some algorithms and a comparison with the pre-cited existing solutions. A conclusion and some perspectives finish this paper.

II. Security in Ubiquitous Systems

Ubiquitous systems are mainly distributed, reactive to context, and deal with user personnel data. It is therefore necessary to give more autonomy to their security systems, mainly by providing them with mechanisms through dynamic and flexible cooperation and collaboration to ensure the smooth flow of data in this system. We must develop robust protocols that ensure high confidence in the services and minimize the vulnerabilities of such systems.

A. *Ubiquitous features*

The main features of ubiquitous environment are the user mobility and the proliferation of light devices, communicating through light and wireless infrastructure. Thus, the convergence of terrestrial infrastructure (Local Area Network LAN, fiber optic, etc.) and mobility (Global system for mobile GSM, 4G and WIFI) enables users to have access to a vast and limitless network of information and services regardless of place and time. All these features create complex security problems. This requires the introduction of advanced authentication methods, the management and distribution of security keys between the various entities on the network, while respecting the constraints of wireless networks, such as the radio interface capacity and mobile devices, resources that represent the bottleneck of such networks.

B. *Security Requirements*

The main issues that must be addressed in terms of security are [10]:

1) Authentication mechanisms and credential management
2) Authorization and access control management
3) Shared data security and integrity
4) Secure one-to-one and group communication
5) Heterogeneous security/environment requirements support
6) Secure mobility management
7) Capability to operate in devices with low resources
8) Automatic configuration and management of these facilities.

To guarantee the security of ubiquitous systems, they must meet the following requirements as defined in [11]:

- Decentralization: Ubiquitous environment is designed to allow the user and all its resources to be accessible anywhere and anytime. The mobile user must have access to his attributes, and prove his identity in this environment without claiming all the time the centralized server of his organization. The security policy implementation should be as decentralized as possible.
- Interoperability: The heterogeneity is a feature of ubiquitous applications. The proposed solution involves the implementation of a decentralized system for collaboration and interaction between heterogeneous organizations.
- Traceability and non-repudiation: The design of a completely secure ubiquitous system is impossible. But, the implementation of mechanisms to quickly identify threats or attacks (such as non-repudiation / tracking) provides an acceptable issue.
- Transparency: Ubiquitous computing aims to simplify the use of its resources. In ubiquitous applications and environments, the problems of authentication are more complex because of the lack of unified authentication mechanism. Several techniques have been designed to make user authentication easy and done in a transparent manner (Single Sign On).

- Flexibility: New authentication techniques have emerged such as biometrics, Radio frequency identification (RFID), etc.. Thus, a security system for ubiquitous environment must be able to integrate these different means of identification and adapt authentication mechanisms to the context of the user, and the capacity and the type of used devices.
- Protection of Privacy: The identity and attributes of a person are confidential information that is imperative to protect. To secure these data we must implement protocols that protect and ensure confidentiality.

III. Privacy in Ubiquitous Systems

The implementation of security solutions in ubiquitous environments has many constraints, like limited capacity of batteries, device mobility and limited time response. Several security systems providing authentication have been proposed and we chose to detail three of them: GAIA, OpenID and UPP.

1) GAIA: GAIA [12] is a system proposed by the University of Illinois which provides an infrastructure to build applications for the ubiquitous environment. GAIA security is integrated as a component of the architecture known as the GAIAOS Security, which includes a central service Cerberus integrating authentication and access control. GAIAOS Security uses the identity as a basis and provides the ability to authenticate the user regardless of the capacity of used devices by using different methods such as conventional mechanisms login / password or smart cards.

2) OpenId: OpenID is an authentication system designed for the World Wide Web. It allows users to authenticate to multiple sites (which support this technology), without having to remember one login for each, but using only once a unique identifier of type URI (Uniform Resource Identifier) which is obtained by one of the OpenID providers [13]. Three main actors are involved in authentication:

- The end user (client)
- The Consumer (Relying Part: RP)
- The identity provider (OpenID Provider: OP / IDP Identity Provider)

The authentication process during a connection is as follows (Figure 1):

1) The client provides URI of his profile OpenID to the consumer site (RP).
2) The RP contacts the OpenID provider's site (OP).
3) Both RP and OP exchange digital information generating a "shared secret"
4) The consumer site (RP) redirects the client in a transparent way to OP.
5) The client authenticates to the provider site (OP).
6) The provider site (OP) redirects the client to the consumer (RP) with the authentication result.

3) User-directed Privacy Protection: The authors in [14] proposed an architecture named User-Directed Privacy Protection (UPP) that allows users to control their information themselves in order to access services. The UPP architecture consists of two parts:

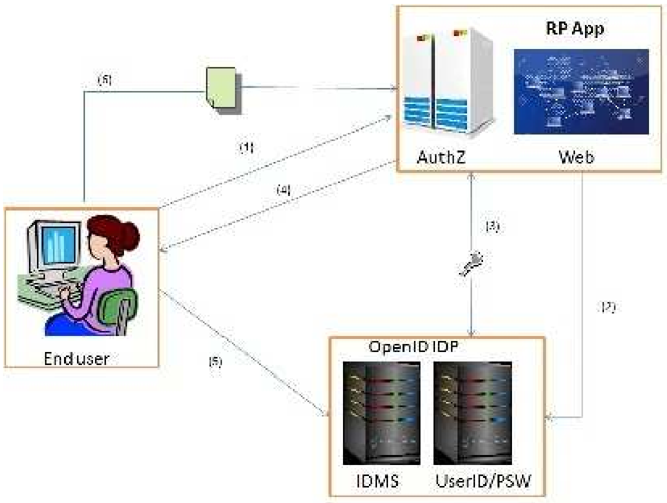

Figure 1: OpenId authentication process [13]

- Authentication Manager: manages the personal information and authentication information.
- Control Center: the part that controls the equipment and contents.

The user uses a personal device (a PDA for example) to authenticate to the security manager of a domain, in order to request a service. The service provider checks if the user manages its list of access information (login, password) himself. If so, before entering the service, he transfers its access policy to the domain manager. The latter "negotiates" the policy with the user and sends this information to a general manager. The authentication process is as follows (Figure 2):

1) Pre-authentication of each security manager of a domain will be established by the manager of global certification.
2) The user authenticates to the security manager of the new domain, it issues a temporary certification to the user by using an encrypted channel.
3) If the user requests a service to the new domain, it requires a new user authentication for access to Global Authentication Manager.
4) The new domain requires a certification to the manager of global certification, and it confirms the request (offer a certificate).
5) The manager of global certification requests personal information to the Global Authentication Manager. The latter provides the necessary information.

A. Synthesis

In this section, we will compare the different approaches based on the requirements of ubiquitous computing security.

- Decentralization: Security systems described here are based on the centralization of user data. Security GAIAOS backups user data in server Cerberus, OpenID has a third party (the identity provider OP) which is the only dedicated server for data storage and users

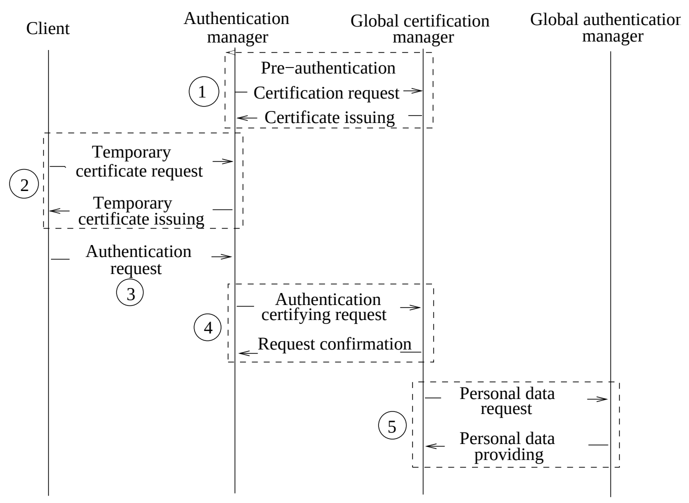

Figure 2: UPP authentication algorithm [14]

authentication, and the manager of the global authentication server is the only server that architecture UPP uses to collect users' certificates.

- Interoperability: Users must have mobile access to the site to which they are affiliated, and to the other remote sites, in order to obtain the desired services. The interoperability of different security policies is essential. OpenID is a great innovation in terms of interoperable solutions for authentication and identity management on the Internet. GAIAOS Security does not support collaboration between organizations, which restricts the security policy to the local domain only.
- Traceability: Traceability is very important; it is based on mechanisms to ensure non-repudiation in order to track any person who committed the fraud. GAIA, OpenID and UPP have the advantage of traceability that allows the account owner to have a history of its subscriptions.
- Transparency: GAIAOS and OpenID provide a way to reduce the interaction of the user during the authentication process. The UPP architecture and the propositions given in [2] and [6] do not provide this feature because users control their information themselves to access services.
- Flexibility: the diversity of devices used in ubiquitous computing ensures that the mobile user has the ability to connect and interact with its environment by using different techniques of identification. GAIAOS, UPP and OpenID provide the ability to authenticate the user regardless of the capacity of the used devices.
- Protection of Privacy: It is necessary to protect the identity and private data of mobile users. OpenID, GAIA and UPP are interesting approaches that allow the protection of private data of mobile users, but has the disadvantage of storing different user privileges on a server that is accessible all the time, this makes these systems vulnerable.

IV. Proposition of a New Management System of Private Data

A. Problem Positioning

The development of Web services, the vast heterogeneity of the connection techniques and conditions of communication (including bandwidth), the proliferation of mobile devices, and the heterogeneity of protocols and their deployment in mobile and ubiquitous computing increase significantly the risks related to the protection of user's privacy. Implemented security policies impose protocols that enable the conservation and management of personal data, and limit their transmissions from mobile devices as well as their movement within the network. This is a good approach to avoid some attacks like sniffing.

The security solutions presented previously are typically based on backing up data on a single server. The private data of the user are stored on a single server, the invocation (request) to a secure service by a user, will acquire its data from the server after an authentication procedure. These solutions however suffer from two deficiencies: the first is the inability to access the data without a reliable connection, secure, permanent and fast server, a set of conditions difficult to meet in any environment. The second is the centralization of data on a single server which represents a vulnerability because if the server is compromised the entire system will be.

As part of our project, we will mainly deal with the following two issues:

- How to protect private data of the mobile user in a transparent way, easily and without being intrusive?
- How to decentralize the data and the user's personal information in a fast and secure manner?

B. The Proposed Architecture

To satisfy ubiquitous environmental security requirements such as decentralization, flexibility and protection of private data, we opted for a hierarchical architecture. The principle of this solution is the distribution of the user data on a set of servers so that each of them contains only the information needed for user authentication and the servers (nodes) are distributed randomly over a virtual structure. This data is scattered in the system as follows:

- Personal data is not on a single server, but on multiple different servers.
- No server owns the totality of a particular client personal data.

The entities involved in this architecture are as follows:

- The user: a human being (client), who is the consumer of the service.
- Generator of identifier (GenID): a node that is responsible for generating a unique identifier for users during their registration in the system.
- Domains: A domain represents a business, a service provider (music, videos, bank, etc.).

The architecture is based on the following hypothesis:

- The architecture is ephemeral and only the request message and the transmission of personal information uses the links.
- No node knows the entire structure.
- A node knows only its successors and its predecessor.
- A pre-authentication of the domains of the environment is performed using a third party authentication.
- Each user has at least one certificate (issued by his domain of affiliation) and can acquire others in other domains.

Each user has a universal identifier, distributed among all domains at its first registration in the system, which allows gathering its data. Some user data can be replicated on some servers, but each of them stores the personal information necessary to it and the additional information obtained from other nodes are deleted.

C. Illustrative Example

Suppose Alice has an identity certificate containing her name, photograph, date of birth, address, Social Security number, fingerprint, account number, her public key and her profession.

$$\begin{array}{c}\text{Alice complete}\\ \text{identity certificate}\end{array} \rightarrow \left\{\begin{array}{l}\text{Alice}\\ \text{Professor}\\ \text{Birth date}\\ \text{Security ID}\\ \text{Photo}\\ \text{Fingerprint}\\ \text{Public key}\\ \text{Address}\\ \text{Bank account ID}\end{array}\right.$$

If she wants to rent a car, Alice must present a document (certificate) confirming the user name and some personal information such as her photo and address. However, the same document may contain other information that Alice does not wish to divulge, such as age or job.

$$\text{Car rental}\rightarrow \left\{\begin{array}{l}\text{Alice}\\ \sout{\text{Professor}}\\ \sout{\text{Birth date}}\\ \sout{\text{Security ID}}\\ \text{Photo}\\ \sout{\text{Fingerprint}}\\ \sout{\text{Public key}}\\ \text{Address}\\ \sout{\text{Bank account ID}}\end{array}\right.$$

This case is not unique. During a consultation with a doctor, Alice must be able to present a document showing only the name, date of birth and social security number. This illustrates the need for mechanisms for the decentralization of personal information in order to protect the private data of users.

$$\text{Doctor}\rightarrow \left\{\begin{array}{l}\text{Alice}\\ \text{Birth date}\\ \text{Security ID}\\ \text{Photo}\end{array}\right.$$

$$\text{Bank}\rightarrow \left\{\begin{array}{l}\text{Alice}\\ \text{Professor}\\ \text{Fingerprint}\\ \text{Bank account ID}\end{array}\right.$$

D. The Broadcast Distributed Privacy Protection System Architecture

To achieve decentralization of private data, we proposed a distributed architecture named Broadcast distributed privacy protection system (BDPPS) based on the decentralization of private data, so a hierarchical architecture is needed. To reflect structural relationships and hierarchies, we used a binary tree. The advantages of binary trees are well known: flexibility, easy construction and management (searching, insertion), etc.

The basics of this architecture are as follows:

- Private user data is distributed on a set of servers so that each one contains only the information necessary for its operation.
- The domains are distributed over the nodes of the tree in a random manner.
- If a domain needs the private data of a user who depends on another domain, a search request will be broadcasted on all system nodes.
- Upon receipt of the response, there is a deadline for the additional data to be deleted.

The major drawback of this architecture is the large number of requests sent through the tree when searching private information. To remedy this problem we decided to improve this proposal, based on how to divide the domains in the system.

V. Improved Solution

To minimize the number of messages circulating in the tree and increase the quality of service, we proposed an improved architecture named Tree Based distributed privacy protection system (TBDPPS). The idea is to increase the probability of finding the sought data by depth-first traversal of the tree, and to arrange these data in a complementary manner with between two close nodes (servers).

The organization of services is done in a manner allowing the users data to be structured in a complementary and easy way. The sent request follows a tree structure in depth in order to increase the probability of finding the researched data. If a server needs more information, instead of asking the user, it retrieves them from the nearest server in the tree. Each node server is supposed to receive a request from a parent node or a

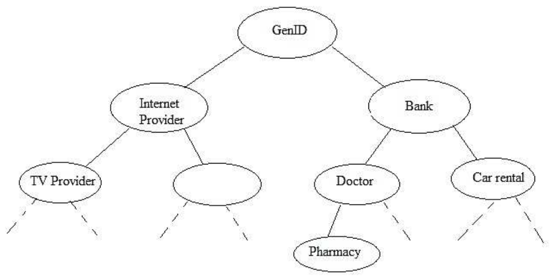

Figure 3: The tree broadcast distributed privacy protection system principle

child node for some specific information that he has but they don't.

For example a service that has an activity like downloading videos, music, etc., it would be better to have the bank node as a closest neighbour, in order to complete the transaction process as quickly as possible (Figure 3).

This distribution of domains offers various advantages:

- Message number Reduction flowing in the tree.
- Increase in the quality of service.
- Simplicity and ease of personal data management.

A. *Example*

Following the previous example, by using her PDA Alice was authenticated with a car rental service to rent a car. According to the proposed architecture, it is the car rental server that will retrieve data about the payment (account Identifier).

The server prepares a query that contains the necessary parameters such as domain code, the user ID and the needed data (Bank account ID), and then sends them to his child nodes on the tree. The latter seeks the ID of the user and the account number, if they have the desired data they send the response request containing the necessary information, if not they send the request to their child nodes and so on. If no child node exist then the request is sent to its parent node. The car rental service node and the bank node belong to the same subtree, as shown in Figure 3.

B. *Algorithms*

In the following section, the different algorithms executed by the tree's nodes are described and they use the following defined variables:

`codD`: The domain code which sends the research request.

`reqID` : Request identifier.

`userID` : User identifier.

`privateData` : The set of private data belonging to a domain.

`Ldata` : the set of sought data by the request issuer.

`data` : the set of data conveyed by requests/responses.

`found` : a Boolean variable (initially `FALSE`).

1) User registration: When a user submits a registration request to a domain in the system for the first time, this domain sends a request to the GenID node. This node first verifies the validity of the request (a real new user), if it is valid it generates a unique identifier (a numeric or alphanumeric string), then broadcasts it on the tree. The algorithm implemented on the GenID node is given in Algorithm 1.

Algorithm 1 User registration

```
Require: Request by a new user
Ensure: User identifier userID.
1: if new user then
2:     if current node code = GenID code then
3:         generate a userID to user
4:     end if
5:     register user to domain
6:     send(codD, reqID, userID) to child nodes.
7: end if
```

2) Service request: When a user requests a service to a domain the latter searches its database to retrieve the user's private data. If there is a lack of information necessary for a proper operation of the service, the server propagates a request containing some parameters in its sub-tree to find the missing data simultaneously. If the answer obtained from its sub-tree is negative then the request will be sent to the parent node.

The search stops when the initiator domain has recovered all the necessary data, or has received the request sent by a node (child for the root node or parent for other domain) and the variable found is false. The main steps are as follows:

Step1: The user submits a service request to a domain as given in Algorithm 2.

Algorithm 2 Service request Algorithm

```
Require: Request by a user affiliated to domain
Ensure: Satisfaction of a service
1: if data $\subset$ privateData then
2:     service satisfied
3: else
4:     send(codD, reqID, userID, data, found) to
       child nodes
5: end if
```

Step2: The receipt of the request by another domain: Upon receipt of this request, the domain checks if the user ID and data exists, if yes it will formulate a response containing the found data and sends to the sender (`codD` of the request), otherwise it sends the request to his child nodes, if they exist, or to its parent node. The result is given in Algorithm 3.

The statement `data`$\leftarrow$`privateData` concerns only the wanted data from the set `privateData`.

Step3: The receipt of the request by the issuer: Upon receipt of the request, the issuer verifies the boolean variable `found` if it is true. Then it compares the data received with the data sought and if all the data are found then the service is executed, otherwise the issuer will make another request by omitting

Algorithm 3 Request reception Algorithm

```
Require: Request(codD, reqID, userID, data, found)
Ensure: Collect missing private data
1:  if (userID $\in$ domain) && (data $\subset$ privateData then
2:      found $\leftarrow$ TRUE
3:      data$\leftarrow$privateData
4:      send(codD, reqID, userID, data, found) to codD node
5:  else
6:      if $\exists$ child nodes then
7:          send(codD, reqID, userID, data, found) to child node
8:      else
9:          send(codD, reqID, userID, data, found) to parent node
10:     end if
11: end if
```

all the found data and sending it to another child if it exists or to the parent to explore another sub-tree. The service is unsatisfied when the issuer receives the request by one of its neighbors (child for the root and parent for other nodes) and the variable found is FALSE. The term ”card” stands for the cardinal of a set. Algorithm 4 illustrates this step.

Algorithm 4 Issuer request reception Algorithm

```
Require: Request(codD, reqID, userID, data, found)
Ensure: Satisfaction of a service.
1:  if found=TRUE then
2:      if card(Ldata) = card(data) then
3:          Service satisfied
4:      else
5:          data $\leftarrow$ Ldata-data
6:          send(codD, reqID, userID, data, found) to child node
7:      end if
8:  else
9:      if parent node not visited then
10:         send(codD, reqID, userID, data, found) to parent node
11:     else
12:         Service not satisfied
13:     end if
14: end if
```

The statement data $\leftarrow$ Ldata-data concerns the case when data contains more than one item, so the found items are retrieved from the set data to continue the search for the rest of items.

If a service is satisfied the Ldata is deleted after a fixed delay. If all the links of the tree exist, then all the needed data exist on the tree and it will be found. In this case the searching time will be at maximum the time of parallel browsing of the tree (height size).

A service cannot be satisfied if the needed data is not found, and this is possible only if the concerned server node (which has the data) or the links are down. In this case a request is repeated after a random delay.

VI. Synthesis

We have proposed a solution that solves the problem of data privacy for mobile users. Our proposal is to define a new architecture that takes into account the separation of different domains in the system and corresponds to a tree. The user's personal data are distributed across a set of servers so that none will ever have all the user's private data except those required for its operation.

- Decentralization: In the proposed system, the different domains making up the ubiquitous environment do not share user's private data. Each domain maintains a subset of the user's necessary data.
- Interoperability: the collaboration between the nodes of the system is done to allow a collection of different private data that a domain needs. Each system node can communicate with other remote nodes across his neighbors, by sending the different requests.
- Transparency: the TBDPPS system reduces the interaction of the user during the authentication process and service request. Indeed, a user authenticates first to a service then can acquire other services in an easy and intuitive way, because it is the first server that will retrieve the rest of user private data.
- Traceability: transactions in our system are made via certificates that guarantee non-repudiation of users (certificates owners) in order to identify any performed transactions.
- Flexibility: The system TBDPPS offers the user the possibility to be authenticated regardless of the capacity of the devices used and the different identification methods.
- Privacy protection: Taking into account the separation of the different data on separate domains of the system, so that an intruder cannot have the totality of the user's private information and protect these data against unwanted disclosure, the proposed architecture allows the protection of users private data and overcomes the problems of their storage on a vulnerable single server.
- Data distribution: The propositions given in [7] and [9] deal with distributed private data but the client is an actor, so no transparency. For the latter, it even preconizes a tree architecture but noisy information are included. In our proposition only the private data is distributed, which means less data transmission.
- Autonomy: The proposed system operates without the client intervention. So a hacker cannot get a user's private data. Attacks like sniffing cannot succeed because only some of private data is circulating on the network. Finally, the only dangerous attack is a non-trusted or corrupted server (node), but we supposed that all the domains are authenticated using a third-party protocol.
- Number of messages: Only one type of message will be used. A request is used to collect the missing private data and the same request is used to send the response to the request issuer.

- Algorithmic complexity: the complexity of the proposed method is given depending on the type of trees (from the best to the worst);

Type of binary tree	Complexity
Complete tree	$O(\log n)$
Full tree	$O(\log n)$
One-branch tree	$O(n)$

and on each situation.

Situation	Complexity
Registration	$O(\log n)$
Full private data present	$O(1)$
One missing item	$O(2 * \log n)$
More than one missing	$O(2 * \log n)$

The variable n is the number of domains/nodes in the system/tree.

- Figure 4 shows the results of a small simulation (using Matlab) of the time response of the proposed method depending on the size of tree and the number of missing items in the data. The time response depends on the tree height ($log(n)$ and, even if the number of missing items increases, the parallel parsing of the tree is done once.

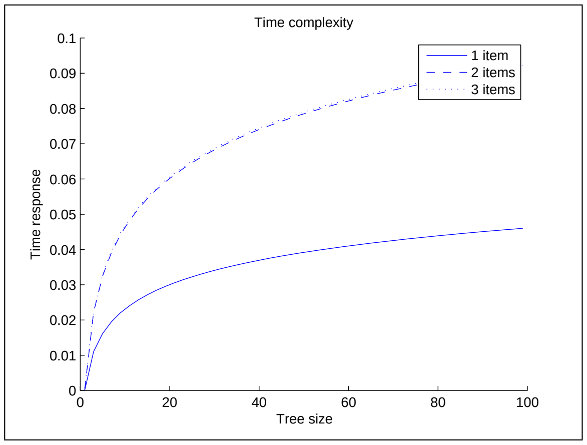

Figure 4: Time response for a single request

VII. Conclusion

Ubiquitous environment allows performing the appropriate actions to the user while adapting to environmental conditions, preferences and user profile. Building such an environment is very difficult given the user's everyday environments composed of heterogeneous devices leading to a dynamic system.

Existing solutions like GAIA, OpenID and UPP systems are very convenient structures for the organization of private data, but the critical point of these projects is the centralization of data on a single server, making data protection of mobile users very vulnerable.

We proposed a solution to protect user's private data which overcomes the aforementioned deficiency and takes into account decentralization and the method of domain dissociation to make communication easy and flexible. The number of domains is limited so the tree size is limited and, since it is a binary tree, its construction will be easier.

As future work, it would be interesting to consider a dynamic construction method of the virtual tree. Only the one-to-one links of the tree are to be built by identifying the parent-child link. This may be done at the first request by the Generator of identifiers node. To achieve this a method for domains dissociation in the system based on private data located in each node would be necessary.

References

[1] M. Weiser, "The computer for the 21st century," Scientific American, vol. 265, 1991, pp. 94–104, ISSN: 0036-8733.

[2] J. Kim, K. Kim, J. Park and T. Shon, "A scalable and privacy-preserving child-care and safety service in a ubiquitous computing environment," Mathematical and Computer Modelling, vol. 55, 2012, pp. 45-57, ISSN: 0895-7177.

[3] J. S. Winter, "Surveillance in ubiquitous network societies: normative conflicts related to the consumer in-store supermarket experience in the context of the Internet of Things," Ethics and Information Technology, vol. 16, March 2014, pp. 27-41, ISSN: 1572-8439.

[4] H. Nissenbaum, Privacy in context: technology, policy, and the integrity of social life. Stanford University Press, Stanford, 2010, ISBN: 0804752370.

[5] C. Dhasarathan, S. Dananjayan, R. Dayalan, V. Thirumal and D. Ponnurangam, "A multi-agent approach: To preserve user information privacy for a pervasive and ubiquitous environment," Egyptian Informatics Journal, vol. 16, 2015, pp. 151-166, ISSN: 1110-8665.

[6] Y. Zuo and T. O'Keefe,"Post-release information privacy protection: A framework and next-generation privacy-enhanced operating system," Information Systems Frontiers, vol. 9, 2007, pp. 451-467, ISSN: 1387-3326.

[7] C. Dong, L. Chen and Z. Wen,"When Private Set Intersection Meets Big Data: An Efficient and Scalable Protocol," in Proceedings of the 20th ACM Conference on Computer and Communications Security (CCS), November 4-8, 2013, Berlin, Germany. ACM Communications, Nov. 2013, pp. 789–800, ISBN: 978-1-4503-2477-9.

[8] N. Shang, G. Ghinita, Y. Zhou and E. Bertino,"Controlling Data Disclosure in Computational PIR Protocols," in Proceedings of the 5th ACM Symposium on Information, Computer and Communications Security (ASIACCS) April 13-16, 2010, Beijing, China. ACM Communications, Apr. 2013, pp. 310–313, ISBN: 978-1-60558-936-7.

[9] J. Zhang, X. Xiao and X. Xie,"PrivTree: A Differentially Private Algorithm for Hierarchical Decompositions," in Proceedings of the International Conference on Management of Data (SIGMOD), June 26–July 01, 2016, San Francisco, CA, USA. ACM Communications, Jul. 2016, pp. 155–170, ISBN: 978-1-4503-3531-7.

[10] J. Al-Jaroodi, I. Jawhar, A. Al-Dhaheri, F. Al-Abdouli and N. Mohamed, "Security middleware approaches and issues for ubiquitous applications," Computers and Mathematics with Applications, vol. 60, 2010, pp. 187–197, ISSN: 0898-1221.

[11] R. Saadi, The chameleon : A security system for nomadic users in collaborative and pervasive environments. PhD Thesis, Institut National des Sciences Appliques de Lyon, France, Jun. 2009.

[12] M. Roman, R. H. Campbell, R. Cerqueiray and C. K. Hess, "Gaia: A development infrastructure for active spaces," Internal document, Department of Computer Science, University of Illinois at Urbana-Champaign, USA, 2001. URL: http://gaia.cs.uiuc.edu/papers/GaiaSubmitted3.pdf [accessed: 2016-05-08].

[13] H. Ludvigsen, L. Valtola, T. W. Kim, H. Shu, A. Johnsen and K. Moen, "Signicat openid," 2008. URL: http://www.idi.ntnu.no/emner/tdt4290/Rapporter/2008/g6.pdf [accessed: 2016-05-08].

[14] J. H. Lee, J. K. Park and S. W. Kim, "User-directed privacy protection in the ubiquitous environment," in Proceedings of IEEE/IFIP International Conference on Embedded and Ubiquitous Computing Dec. 17–20, 2008, Shangai, China. IEEE, Dec. 2008, pp. 37–42, ISBN: 978-0-7695-3492-3.