Tree Based Distributed Privacy 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*—Ubiquitous computing aims to integrate computer technology in man's everyday life in various fields. To improve interactivity, it offers the user the ability to access various features and services of its environment and from any mobile lightweight autonomous device while adapting them to the user's context. Cloud computing allowed ubiquitous systems to be more efficient at a more reduced cost. This accentuates security problems and particulary privacy preserving. The existing mechanisms and solutions are inadequate to address new challenges. 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 existing systems. Furthermore, it takes into account the domain dissociation property, in order to achieve decentralized data protection.**

Keywords–Ubiquitous Computing; Cloud 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.

The importance of security and privacy in ubiquitous and pervasive systems is universally agreed. This paper is an extension of initial work in this area that was presented in [1] (UBICOMM 2016). The scope has been broadened and significant extensions has been made. In particular, we have added new material to Section III, Section V, and Section VI. Other amendments have been made throughout the paper.

The origins of ubiquitous systems date back to 1991, when Mark Weiser [2] 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 his environment and from any mobile device (personal digital assistant PDA, tablet computer, smartphone, etc.).

The most important feature of pervasive computing is context awareness. The user context affects the available services as the surrounding networking environment adapts to the needs of the user. Various pieces of information are made available to the networks in order to provide a concise user experience, thus leading to privacy issues.

Cloud computing is another emerging technology that is still unclear to many security problems [3]. Cloud Computing is a model of computing, in which the users can rent infrastructure, platform or software services from other vendors without requiring the physical access to the rented service. There are three main types of cloud offerings: Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS).

IaaS offers virtualized instances of bare machines leaving the installation and customization of softwares including the Operating System to cloud computing customers. In PaaS, an application framework is provided to the customers for developing their software with. A SaaS provider offers a particular application as a web service, which customers can customize to their needs.

The Cloud Service Provider (CSP) focuses on infrastructure and software expertise, and aims to optimize their utility by providing centralized services for one or many clients. The benefit to the cloud service client (CSC) is that the cost associated with the underlying infrastructure and software services, needed to support the CSCs application, is reduced. In spite of the widespread adoption, organizations are still wary of storing their sensitive data with a CSP. Privacy risk remains a major concern in the cloud computing environment.

The emergence of these technologies 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. Centralized approaches are suitable for systems with fewer number of (web) services and limited number of client requests, since it is always prone to bottleneck delays and single point failure. It is therefore necessary to give more autonomy to security systems, mainly by providing them with mechanisms establishing dynamic and flexible cooperation and collaboration.

Privacy is one feature that must be accounted for in all systems that include human users or any kind of data pertaining to humans. This must be planned for, from the design phase, and handled in all phases of system deployment. Privacy is, however, also a difficult concept and largely a culturally dependent trait. What can be expect to keep private, and not the least, from whom do we keep information private. Nevertheless, whatever privacy level we decide on, one should ensure that it is credibly maintained [4].

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. Even though the study has been done for ubiquitous systems, the proposed method can be applied to cloud computing as well.

The paper is organized as follows: after this introduction, some existing research works on the domain are presented in Section II (Ubiquitous environment security requirements) and Section III with a comparison between them. Then, in Section IV, the proposed system is given with an illustrative example. An improved solution based on a tree structure is presented in Section V, 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 and Cloud computing

Ubiquitous systems are mainly distributed, reactive to context, and deal with user personal 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

Different kinds of terms, such as ambient intelligence, ambient networking and ubiquitous computing, have been introduced to portray the visions of enhanced interaction between the users and the surrounding technology. 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.

One vision, preached by [5], lists the following as key requirements:

- Unobtrusive hardware
- Seamless communication
- Dynamic and distributed device networks
- Natural feeling human interfaces
- Dependability and security

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. Cloud computing

There are a variety of ways that the privacy of data can be compromised in a cloud service environment [6]. This includes the following:

1) Sharing of data with an unauthorized party: The Cloud provider could compromise the confidentiality of the data by sharing the data that it stores with unauthorized parties.
2) Corruption of data stored: The Cloud Computing providers root access to physical machines allows them to have access capacity for data modification or deletion.
3) Malicious Internal Users: The employee of a Cloud Computing Provider who has root access to these physical machines, could access the data and use it for their own advantage.
4) Data Loss or Leakage: When a virtual machine is used in an infrastructure, it poses a variety of security issues, which could lead to a compromise of the data.
5) Account or Service Hijacking: If the service is hijacked, or the computer is hacked into by an intruder, the hacker will have access to data.

Storing the data in the cloud, can increase the privacy risks for not only the cloud client (simple or organization) but also for the cloud implementers, the services providers and the data subject. Privacy Enhancing Technologies (PET) can be used by the developers of the application to enhance the individuals privacy in an application development environment. PET technologies include:

- Privacy management tools that enable inspection of server-side policies that specify the permissible accesses to data
- Secure online access mechanisms to enable individuals to check and update the accuracy of their personal data
- Anonymizer tools, which will help users from revealing their true identity by not revealing the privately identifiable information to the cloud service provider.

A state of the art of Privacy solutions in the cloud is given in [3] and [6].

C. Security Requirements

The main issues that must be addressed in terms of security are [7]:

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 and the cloud, they must meet the following requirements as defined in [8]:

- 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. A decentralized approach is always desired whenever dealing with a consequent number of spread data and clients.
- 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, 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. Imposing Privacy in the cloud is still a challenge.

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 [9], 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 [10], the author uses the framework of contextual integrity related to privacy, developed by Nissenbaum in 2010 [11], 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.

Information availability is already evident in the emergence of social networking and the way people freely give out information about themselves and the people they know, providing avenues for identity theft. Thus, in the advent of ambient computing environment, user has to trust the system in order to agree to disclose information about themselves, i.e., adjust their privacy settings accordingly. However, the trust evaluation made by a person can be affected and it is not always a rational thing.

Trust is a concept that may involve and justify the disclosure of personally identifiable sensitive information. Trading privacy for trust is thus a way for balancing the subjective value of what is revealed in exchange of what is obtained. A flexible privacy-preserving mechanism trading privacy for trust-based and cost-based incentives is given in [12]. In a classical view of privacy, a user exposes (part of) personal information in order to be trusted enough to get access to the service of interest. In other words, privacy disclosure is traded for the amount of reputation that the user may need to be considered as a trustworthy partner in some kind of negotiation.

Mobile terminals are usually personal items, so privacy is to be considered when virtualization in cloud computing is used and data processed remotely. In [13], a mobile terminal virtualization framework is proposed to meet issues such as security, privacy and quality of service by encrypting data communications by the cloud server usign an asymmetric cryptography scheme.

The author of [14] presents a study of privacy implications of location-based information provision and collection on user awareness and behaviour, in the particular case when using GeoSNs (Geo-Social Networking applications). The first result is the extent of potential personal information that is derived from location information, and the second result is the need to improve users knowledge, access and visibility of their data and to be able to control and manage their location data.

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 [7], some representative security middleware approaches are reviewed and their various properties, characteristics, and challenges are highlighted.

Privacy by Design concept integrates respect for users privacy into systems managing user data from the early stage. Mobile applications do not suit this concept and lack transparency, consent and security. In [15], a new permission model suitable for mobile applications is given. It is integrated into mobile operating systems; well designed, it makes a proactive privacy-respecting tool embedded in the system. The authors

focus permissions on data and the action that can be carried out on this data, rather than on the technology used.

A. Literature Review

Several security systems providing protection of sensitive data have been proposed and we chose to detail some of them:

1) Hybrid Hash-based Authentication (HHA): Dhasarathan et al. [16] 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).

2) Privacy-enhanced Operating Systems (POS): In [17], 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.

3) Private Information Retrieval (PIR): This protocol allows 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 [18], 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.

4) Private Set Intersection (PSI): Efficiency and scalability become critical criteria for privacy preserving protocols in the age of Big Data. In [19], a new Private Set Intersection 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.

5) Differential Privacy: 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 [20] 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.

6) Paillier scheme: Nowadays, biometric data are more and more used within authentication processes. Such data are usually stored in databases and underlie inherent privacy concerns. Therefore, special attention should be paid to their handling. The most currently available biometric systems lack sufficient privacy protection. The authors of [21] propose a privacy preserving similarity verification system based on the Paillier scheme. This scheme, being an asymmetric as well as additive homomorphic cryptography approach, enables signal processing in the encrypted domain operations. They also introduce a padding approach to increase entropy for better filling the co-domain, combine the benefits of signal processing in the encrypted domain with the advantages of salting. The concept of verification of encrypted biometric data comes at the cost of increased computational effort. The proposed scheme in [21] lowers the error rates and reduces the amount of data disclosed in an authentication attempt using a privacy-preserving biometric authentication scheme.

7) Pseudonymization: Pseudonymization as a data privacy concept is not new and in general it is about who creates the pseudonyms, who has access to them and who has access to data. In [22], the author presents a unified view on pseudonyms and an in-house pseudonymization solution. A pseudonym is a local identifier with no relation with the demographics of a person. Persistent identifiers are introduced to maintain the updates and internal matching considerations. Then an algorithm, to create a pseudonym from a person identifiers, is given, with a national pseudonymization service to resist update problems and wrong matching decisions.

8) Chaavi: A privacy preserving architecture as a solution for webmail systems is given in [6], in which users can retain their mail in the servers of their service providers in a cloud, without compromising functionality (searchability of mails) or privacy. The authors proposed *Chaavi*, a webmail infrastructure, based on the public/private key model, to encrypt email with a custom implementation of encrypted indices for keyword searches, using the servers infrastructure. Chaavi consists of the following components:

- A browser: The browser is responsible for rendering the pages created by the web application.
- Browser extensions: They are used to encrypt the secure message sent to the server, to decrypt the messages that are sent from the server and, additionally, they have key generation and key management functionality.
- A Web application: The webmail application provides graphical user interfaces for the users to read, send and search messages.
- A data base: This database is looked up when the user performs a keyword search.

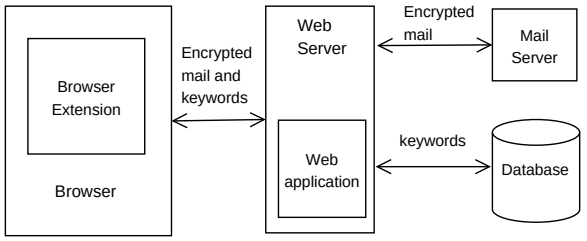

Figure 1: Chaavi - Architecture [6].

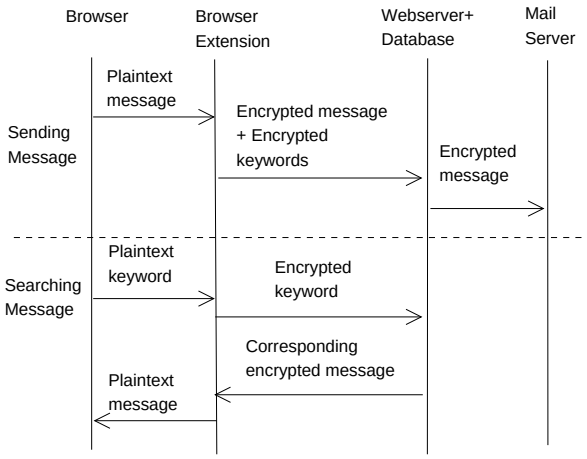

Figure 2: Sending and Searching for a Message in Chaavi [6].

- A mail server: The mail server sends and receives email communicated to it through the Internet.

Figure 1 gives the overall architecture of the system.

When a user sends a message from the web application (Figure 2), the Encryption module encrypts the message and extracts and encrypts the keywords. The web application sends the encrypted message and keywords to the web server. On receiving the encrypted message and the keywords, at the server-side, the application saves the encrypted message alongside with the encrypted keywords in a database for future retrieval. The application then transfers the mail to the Mail Server (SMTP server) to be be delivered to recipient.

9) TREMA: Trust of a peer is based on its prior transactions with other peers. The main challenge is how to collect and distribute reputation scores of peers efficiently. TREMA [23] is a tree-based reputation management solution where nodes are organized at different positions in a tree, based on their reputation, with peers of higher reputation at higher levels. A peer always trusts his ancestors while he is answerable for his descendants. When two peers execute a transaction, a trust route is formed between them. If the transaction succeeds a reward is given to all nodes in the route, but if the transaction fails all the nodes in the route are penalized. If a child becomes trustee than his parent, a swap of their positions is done. One inconvenient in using a tree structure is the possibility to obtain a weakly connected tree, which may cause a partition. the authors proposed adding extra-links. The implementation proposed is based on the following APIs:

- NodeFind: finding and connecting a new node to an existing one in the system.
- NodeJoin: a new node that wishes to join the network, NodeFind must be executed first then a message ”JOIN” is sent to the contact node. If the contact node is not the correct one, it forwards the message to the nodes in its subtree. This opertaion may take $O(logn)$ steps.
- NodeLeave: If a node wishes to leave the network, then the system will establish new tree links and close old ones.
- NodeFailureDiscovery: In case a node discovers one of its neighbors is not responding, then the node will be considered as a ”leave node” and NodeLeave will be called.

10)iPrivacy: Users wish to preserve full control over their sensitive data and cannot accept that is accessible by the service provider. Previous research was made on techniques to protect data stored on un-trusted servers. An approach where confidential data is stored in a highly distributed data base, partly located on the cloud and partly on the clients, is given in [24]. To ensure data protection three major techniques on managing sensitive data exist:

- Data encryption
- Data fragmentation and encryption
- Data fragmentation with owner involvement.

These approaches suppose that the data is stored uniquely on cloud servers. The author of [24], proposes that the user gets a copy of data and a local agent maintains authorized data replicated and accessed by other authorized users in the cloud. The solution consists of two parts: a trusted client and a remote untrusted synchronizer (see Figure 3). The client maintains local data storage where:

- The files that she owns are (or at least can be) stored as plain-text;
- The others, instead, are encrypted each with a different key.

The Synchronizer stores the keys to decrypt the shared dossiers owned by the local client and the modified dossiers to synchronize. When another client needs to decrypt a dossier, she connects to the Synchronizer and obtains the corresponding decryption key. The data and the keys are stored in two separate entities, none of which can access information without the collaboration of the other part.

B. Synthesis

The different approaches have been evaluated based on the requirements of ubiquitous computing security (see Table I where ✓ stands for requirement guaranteed by the method, X otherwise, and − means that the requirement is not relevant).

- Hybrid Hash-based Authentication: The solution is based on multi-agents in cloud environment so decentralization and interoperability is evident. Transparency and Flexibility are not guaranteed because

TABLE I: COMPARISON BETWEEN EXISTING PRIVACY SOLUTIONS.

	Decentralization	Inter-operability	Traceability	Autonomy	Flexibility	Privacy
HAA	✓	✓	–	X	X	X
POS	✓	–	✓	X	X	X
PIR	–	–	–	X	X	✓
PSI	✓	✓	–	X	X	✓
Differential Privacy	✓	–	–	–	X	✓
Pallier Scheme	X	X	✓	X	X	✓
Pseudonymisation	✓	✓	✓	X	X	X
Chaavi	X	X	✓	X	X	X
TREMA	✓	✓	✓	X	X	X
iPrivacy	✓	X	✓	X	X	✓

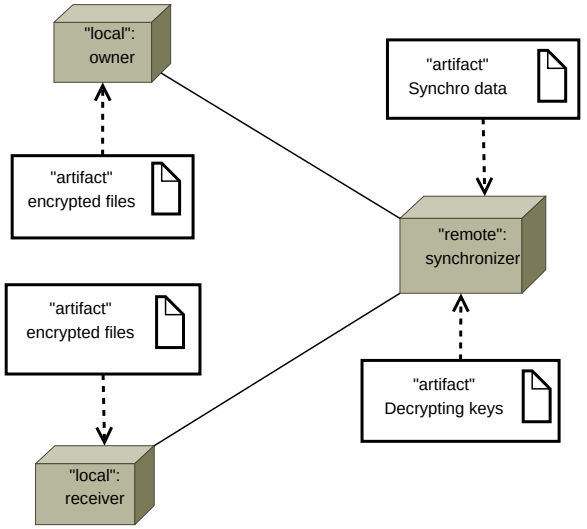

Figure 3: Proposed data management in iPrivacy [24].

the solution is an end-point solution and computations are needed. So we considered that privacy is not guaranteed because intrusion is always possible.

- Privacy-enhanced Operating Systems: The proposed Operating System offers decentralization and inter-operability because it is an OS. But no autonomy or flexibility is offered because the user executes the privacy protection mechanisms. We deduce that privacy is not guaranteed because it is a post-release solution.
- Private Information Retrieval: In this case we cannot talk about interoperability, traceability, or decentralization. But the protocol is not transparent or flexible because the client system takes part in the computations but in the same time this guarantees privacy, because only one item is treated in PIR.
- Private Set Intersection: The protocol treats the case of big data (cloud) so many servers are considered (decentralization and interoperability). Like for PIR protocol, the client takes part in the computations, so it is not transparent or flexible. Privacy is supposed guaranteed.
- Differential Privacy: It deals with data decomposition so decentralization is possible, but the proposed system is not a protocol so interoperability, traceability, or autonomy cannot be evaluated. Because the computations are complex, flexibility is not considered, but this guarantees privacy.
- Pallier scheme: Dealing with biometric authentication mechanism means centralization and homogeneity. The proposed solution is complex, which makes it not flexible but privacy is assured.
- Pseudonymization: Multiple virtual identities means a decentralized supposed inter-operable system. The pseudos are generated by the client application, which makes it not autonomous and non flexible. Traceability is a requirement, for matching considerations, but this also makes privacy not guaranteed.
- Chaavi: It consists of a homogeneous webmail infrastructure with a centralized mail server. The contribution is in the encryption module added to the client browser, which makes it non flexible. Of course, traceability is guaranteed, but not privacy because it is based on a simple messages encryption approach.
- TREMA: The solution is proposed for a Peer to Peer (P2P) system organized as a tree, this implies decentralization and inter-operability. It is based on trust relation between the nodes, so traceability is also supposed. But the trust management and the possible change of position in the tree, makes it not flexible and lacks autonomy. We supposed that privacy is not guaranteed because it is a trade-off between trust and privacy.
- iPrivacy: The system supposes a highly distributed database, which means decentralization but no inter-operability. This database is partly located on the cloud and partly on the client, which means no flexibility and no autonomy. Privacy and traceability are of course guaranteed because of the structure of a database.

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. However, these solutions 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.

Alice's complete identity certificate →
- Alice
- Professor
- Birth date
- Security ID
- Photo
- Fingerprint
- Public key
- Address
- Bank account ID

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.

Car rental →
- Alice
- ~~Professor~~
- ~~Birth date~~
- ~~Security ID~~
- Photo
- ~~Fingerprint~~
- ~~Public key~~
- Address
- ~~Bank account ID~~

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.

Doctor →
- Alice
- Birth date
- Security ID
- Photo

$$\text{Bank} \rightarrow \begin{cases} \text{Alice} \\ \text{Professor} \\ \text{Fingerprint} \\ \text{Bank account ID} \end{cases}$$

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.

Fragmentation and distribution of sensitive data has always been the best solution to protect these data (with encryption) in any domain. In [25], a multi-dimensional binary search tree is adopted to adapt geometrical constraints, thus reducing amount of computations in trying to improve the data-mining k-means algorithm for cluster analysis. A binary partition tree is used in [26] as a region-based to process multi-dimensional Synthetic Aperture Radar (SAR) data. In [27], an m-branch tree ($m > 3$) is preferred than binary or ternary trees, to implement a scalable authenticated dynamic group key exchange protocol.

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 ”parallel” depth-first traversal of the tree, and to arrange these data in a complementary manner 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 searched 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 child node for some specific information that it has but they do not.

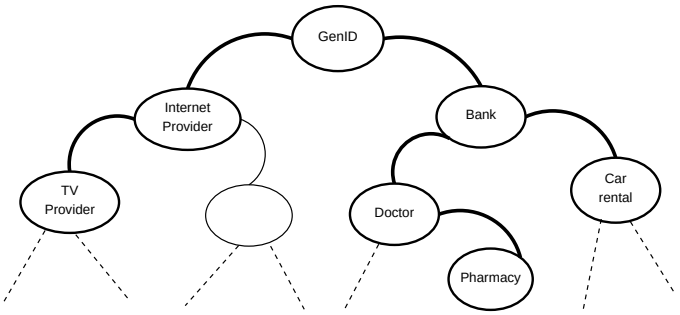

Figure 4: The tree broadcast distributed privacy protection system principle.

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 4).

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 4.

B. Decentralized system structure

The system consists of a set of nodes, which are distributed in a decentralized manner; each node in the system does not have a global knowledge about other nodes except direct neighbours. A tree structure is good for storing and retrieving data.

Definition 1: The decentralized system can be formalized as $T = \{N, L\}$, such that $N = \{n_1, n_2, ..n_m\}$ represents the list of nodes in the network, n_i represents the ith node in the system and m is the total number of nodes, and $L =$

$\{n_i, n_j\}, (1 \leq i \leq m$ and $1 \leq j \leq m$ where $i \neq j)$ is a set of links between different nodes in the system.

Definition 2: Each node n_i in the set N can be formalized as a tuple of $\{S_i, Ln_i, Rn_i\}$ where:

- S_i is the list of services in the ith node
- Ln_i is the node connected to n_i on the left
- Rn_i is the node connected to n_i on the right.

Remark: The GenID node (root) is a particular node and maintains another parameter Ds, a set of all nodes data descriptions Ds_i, which contains the data categories of each node.

The following definition sets the parameters needed to construct the virtual tree based on the user's personal data placed on each node and their level of importance.

Definition3: $D_i = \{d_1, d_2, ...d_k\}$ is a list of user's information affiliated to ith node, for each data, a sensitivity level s_i is associated.

C. Community construction

Each node that is part of the system is considered as an entry point. Each node n_i is connected, at least, to one node n_j that is already present in the system. The establishment of connection between both n_i and n_j is based on the intersection of the sets of sensitive data (same level) needed by both nodes.

The GenID node is created first with the implementation for the system, then each new domain is added to the tree through the root at the request of the service. Joining or leaving the tree will be done by executing the following APIs:

-FindPosition: Finding a node to connect a new one. The best node position will depend on the number of common sensitive items needed by the new node with the existing node. For example, node Pharmacy have much more common items with node Doctor rather than Bank node. Let a new node has Ds_{new} as a data description set of its sensitive data, then the best node to which to connect the new node is the node n_i with Ds_i such that $Ds_i \bigcap Ds_{new}$ is the largest and Ln_i or Rn_i is null. If many nodes satisfy this equation then the node, which will generate a less set of transformations of the tree, will be chosen.

-JoinTree: When a new node wants to join the system, a request is sent to GenID (root), which will execute FindNode to find the best position, then the joining operation is executed (updating tree links).

-LeaveTree: When a node wants to leave the system, a request is sent to the root, and the leaving operation is executed. A node leaves the system if the business or service associated to the domain/node exits no more for example. This operation is critical because some needed data items shared with other nodes may disappear.

- NodeFailure: When a node detects a failed node (non-responding) it sends a request to the GenID node, which will execute the leaving process.

If the tree is skewed and unbalanced the search cost may increase. In a weakly connected tree structure partitions may appear, so extra-links, with non-affected nodes, may be added to reserve places for eventual servers joining the tree.

D. Algorithms

In the following section, the different algorithms executed by the tree's nodes when receiving user's requests 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 needed data to satisfy the user's request.

`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 registration of the new user on the requested service domain will concern only the partial needed private data. If the user is known to GenID but not to the domain, thus a new domain, then it will be registered in this domain with the partially needed private data.

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:    save userID
6:    send(codD, reqID, userID) to child nodes.
7: else
8:    register user to domain
9: end if
```

A user request for a service in a domain will, eventually, lead to its registration in other domains (if not already done), if the fulfillment of the service requires other data associated to these other domains.

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 through both right and left child nodes. 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(Ldata)
Ensure: Satisfaction of a service
1: **if** `Ldata` $\subset$ `privateData` **then**
2: service satisfied
3: **else**
4: `send(codD, reqID, userID, Ldata, 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 (`private data'` is a part of `private data`) and sends to the issuer (`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.

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**

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 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.

The statement `data` $\leftarrow$ `Ldata-data` concerns the case when `data` contains more than one item, so the found items

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**

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, which is the time needed for the service to be satisfied. Each transmitted sensitive data d_i will be accompanied with a time to live (TTL) depending on its sensitivity level s_i.

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. VALIDATION

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.

A. Simulation results

Figures 5 and 6 show the results of a small simulation (using Matlab) of the time response and the number of visited nodes 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 number of visited nodes, which depends on the tree height ($log(m)$) and, even if the number of missing items increases, the parallel parsing of the tree is done at maximum once.

B. Synthesis

The proposed method is also evaluated based on the requirements of ubiquitous computing security as defined in Section II.C:

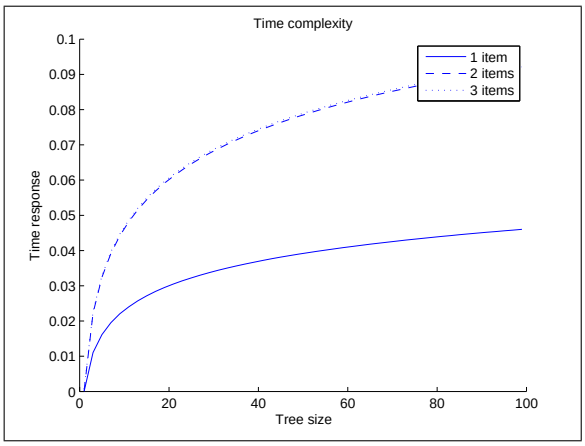

Figure 5: Time response for a single request.

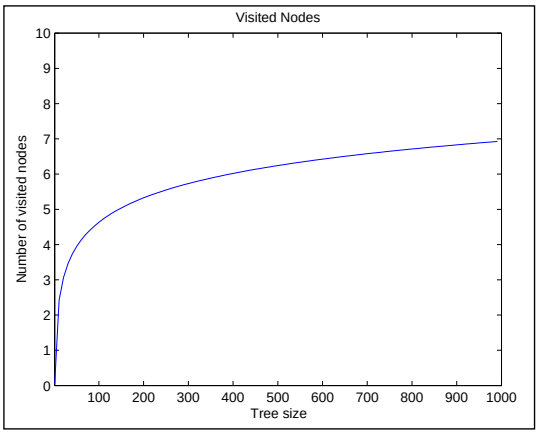

Figure 6: Number of visited nodes.

- 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 the user's 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 use device 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, thus protecting 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 [19] and [20] deal with distributed private data but the client is an actor, so transparency is not verified. 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 trusted 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), and on each situation.

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

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

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

C. Threat analysis

Threat analysis is an important part in security engineering and it forms the basis for the security design of a system. In our threat analysis, we consider following information items to be of special sensitivity: user identity, user contact information, and user bank information.

The main goal for attacks, which we assume in our analysis, is to obtain private information about the user. The threats are considered to be related to illegal combining of user records in different parts of the system, or to the threats introduced by direct external eavesdropping and active intrusion into system

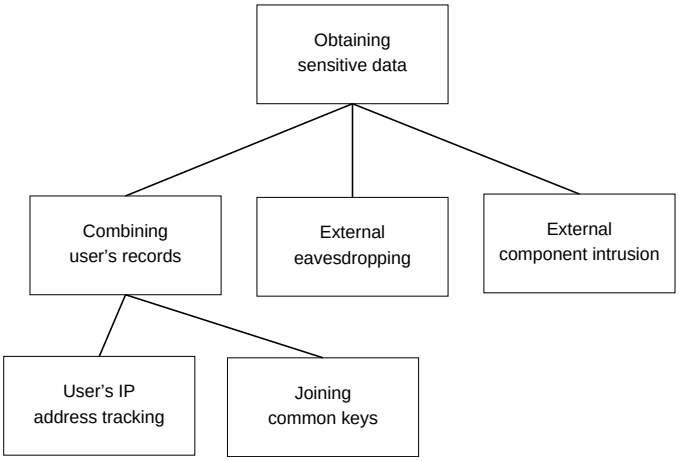

Figure 7: Attacks tree.

components. The attack tree used in our analysis is shown in Figure 7 [28].

The solutions to the different attacks are present in the proposed distributed solution.

- In some cases the IP address may be linkable to a specific device. The private data (not all) is transmitted from the user to a server only once during registration. Then the other transmissions are done between trusted encrypted communicating servers. That's why the second attack "Joining common keys" is also not present. A commonly used method to protect against the threat "user IP tracking" is the use of an anonymities proxy if needed.
- Distributing the user information in the system decreases the impact and the risk of the threat "combining user's records". The distribution may decrease the client privacy concerns, as no one in the system has information about users files, real identity and credit card information.
- As all communications between the user's device and the system and between the different servers of the system are supposed protected using encryption for example, then the probability of successful eavesdropping attacks is very low. The eavesdropping will not impact user's privacy perceptions so much, that it would have a negative impact on the adoption of the proposed solution.
- To avoid intrusion, each user is identified once and affiliated to the first domain (service) requested. This operation is supposed associated with guarantees as for a classical registry in a bank for example. So any intruder will be at least detected by the supposed affiliated domain.

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.

The proposed solution considers a distribution of user's personal data on a set of servers (domains/nodes) linked in a binary tree-based virtual architecture. Examples of such tree are given, and algorithms implementing the registration of a new user and the propagation of a request and its response are proposed.

The proposed method 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. The proposed approach is applicable to ubiquitous systems, but also to cloud computing. Indeed, the different cloud service providers are the domains/nodes of the tree and a user is the cloud service consumer. The communications between the servers are supposed encrypted.

Solutions for the recognized privacy threats leads to some complex security implementations, and a tradeoff between the two is advised, because if users find the system too complex to use, they might find it hard to trust and not adopting it. Distributed solution may require more privacy statements, service agreements, and other legal documents. Searching separated data means more complicated data storage system and data structures in the research analysis.

A dynamic construction of the virtual tree is preconized. 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 user's 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 is proposed. The established communications at the request will be deleted after to obtain a virtual or ephemeral tree.

As future work, it would be interesting to consider a virtual identifier to guarantee confidentiality. Hiding user's identity, by protecting his personally identified information (PII) thus assuring confidentiality, is the first step to guarantee privacy. This approach is our current research work.

References

[1] M. Yaici, S. Ameza, R. Houari, and S. Hammachi, "Private data protection in ubiquitous computing ," in Proceedings of the 10th IARIA Conference on Mobile Ubiquitous Computing Systems, Services and Technology (UBICOMM), October 9-13, 2016, Venice, Italy. pp. 1–8, ISBN: 978-1-61208-065-9.

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

[3] S. N. Kumar and A. Vajpayee, "A Survey on Secure Cloud: Security and Privacy in Cloud Computing," American Journal of Systems and Software, Vol. 4, No. 1, 2016, pp. 14–26, ISSN:1942-2636.

[4] G. H. Køien, "Reflections on Evolving Large-Scale Security Architectures," International Journal on Advances in Security, Vol. 8, No. 1 / 2, 2015, pp. 60–78, ISSN:1942-2636.

[5] K. Ducatel, M. Bogdanowicz, F. Scapolo, J. Leijten, and J-C. Burgelman, "Scenarios for ambient intelligence in 2010," ISTAG report, European Commission, 2001. https://cordis.europa.eu/pub/ist/docs/istagscenarios2010.pdf (last access 06/05/2017).

[6] K. Ramachandran, H. Lutfiyya, and M. Perry, “A Privacy Preserving Solution for Webmail Systems with Searchable Encryption,” International Journal on Advances in Security, Vol. 5, No. 1 / 2, 2012, pp. 36–45, ISSN:1942-2636.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

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

[12] A. Aldini, “A Framework Balancing Privacy and Cooperation Incentives in User-Centric Networks,” International Journal on Advances in Security, Vol. 8, No. 1 / 2, 2015, pp. 16–27, ISSN:1942-2636.

[13] T. Zheng and S. Dong, “Terminal Virtualization Framework for Mobile Services,” International Journal on Advances in Security, Vol. 8, No. 3 / 4, 2015, pp. 109–119, ISSN:1942-2636.

[14] F. S. Alrayes and A. I. Abdelmoty, “No Place to Hide: A Study of Privacy Concerns due to Location Sharing on Geo-Social Networks,” International Journal on Advances in Security, Vol. 7, No. 3 / 4, 2014, pp. 62–75, ISSN:1942-2636.

[15] K. Sokolova, M. Lemercier, and J.-B. Boisseau, “Respecting user privacy in mobiles: privacy by design permission system for mobile applications,” International Journal on Advances in Security, Vol. 7, No. 3 / 4, 2014, pp. 110–120, ISSN:1942-2636.

[16] 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.

[17] 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.

[18] 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.

[19] 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.

[20] 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.

[21] V. Köppen, C. Krätzer, J. Dittmann, G. Saake, and C. Vielhauer, “Impacts on Database Performance in a Privacy-Preserving Biometric Authentication Scenario,” International Journal on Advances in Security, Vol. 8, No. 1 / 2, 2015, pp. 99–108, ISSN:1942-2636.

[22] U. Rothe, “A Generalized View on Pseudonyms and Domain Specific Local Identifiers,” International Journal on Advances in Security, Vol. 7, No. 3 / 4, 2014, pp. 76–92, ISSN:1942-2636.

[23] Q. H. Vu, “TREMA: A Tree-based Reputation Management Solution for P2P Systems,” International Journal on Advances in Security, Vol. 4, No. 3 / 4, 2011, pp. 163–172, ISSN:1942-2636.

[24] E. Damiani, F. Pagano, and D. Pagano, “iPrivacy: A Distributed Approach to Privacy on the Cloud,” International Journal on Advances in Security, Vol. 4, No. 3 / 4, 2011, pp. 185–197, ISSN:1942-2636.

[25] G. Di Fatta and D. Pettinger, “Dynamic Load Balancing in Parallel KD-Tree k-Means,” in Proceedings of 10th IEEE International Conference on Computer and Information Technology (CIT 2010), June 29-July 1, 2010, Bradford, West Yorkshire, UK.. IEEE Computer Society, 2010, pp. 2478–2485, ISBN: 978-0-7695-4108-2.

[26] A. Alonso-González, C. López-Martínez, P. Salembier, S. Valero, and J. Chanussot, “Multidimensional SAR Data Analysis Based on Binary Partition Trees and the Covariance Matrix Geometry,” in Proceedings of 2014 International Radar Conference (Radar2014), October 13-17 2014, Lille, France.. pp. 1–6, ISBN: 978-1-4799-4195-7.

[27] Z. ZeQuan and X. QiuLiang, “Scalable Authenticated Dynamic Group Key Agreement Based on Multi-Tree,” in Proceedings of third International Symposium on Electronic Commerce and Security, 29-31 July 2010 , Guangzhou, China. IEEE Computer Society, 2010, pp. 126–130, ISBN: 978-1-4244-8231-3.

[28] H. Kokkinen, M. V. J. Heikkinen, and M. Miettinen, “Post-Payment Copyright System versus Online Music Shop: Business Model and Privacy,” International Journal on Advances in Security, Vol. 2, No. 2 / 3, 2009, pp. 112–128, ISSN:1942-2636.