International Journal of Research
Available at https://edupediapublications.org/journa
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 03 Issue 04
February 2016

Effective Proposal of a Cloud Assisted Privacy Preserving Mobile Health Monitoring System

Arra Varsha Reddy[1]; B.Manasa[2]& Prof.Dr.G.Manoj Someswar[3]

[1]M.Tech. (CSE), AMR Institute of Technology (Affiliated to JNTUH), RR District, Telangana, India.
[2] M.Tech. (CSE), Associate Professor & HOD, AMR Institute of Technology (Affiliated to JNTUH), RR District, Telangana, India.
[3]Principal & Professor, Department of CSE, NRI Institute of Technology (Affiliated to JNTUH), Greater Hyderabad, Telangana, India.

ABSTRACT:

Cloud-assisted mobile health (mHealth) monitoring, which applies the prevailing mobile communications and cloud computing technologies to provide feedback decision support, has been considered as a revolutionary approach to improving the quality of healthcare service while lowering the healthcare cost. Unfortunately, it also poses a serious risk on both clients' privacy and intellectual property of monitoring service providers, which could deter the wide adoption of mHealth technology. This research paper is to address this important problem and design a cloud assisted privacy preserving mobile health monitoring system to protect the privacy of the involved parties and their data. Moreover, the outsourcing decryption technique and a newly proposed key private proxy re-encryption are adapted to shift the computational complexity of the involved parties to the cloud without compromising clients' privacy and service providers' intellectual property. Finally, our security and performance analysis demonstrates the effectiveness of our proposed design.

KEYWORDS: National Institute of Standards and Terminology (NIST); Infrastructure-as-a-Service (IaaS); Platform-as-a-Service (PaaS); Software-as-a-Service (SaaS); Parkinson's disease (PD); Unified Parkinson's Disease Rating Scale (UPDRS); Resource Pooling

INTRODUCTION

Cloud computing is the use of computing resources (hardware and software) that are delivered as a service over a network (typically the Internet). The name comes from the common use of a cloud-shaped symbol as an abstraction for the complex infrastructure it contains in system diagrams. Cloud computing entrusts remote services with a user's data, software and computation. Cloud computing consists of hardware and software resources made available on the Internet as managed third-party services. These services typically provide access to advanced software applications and high-end networks of server computers.[1]

Figure 1: Structure of cloud computing

The goal of cloud computing is to apply traditional supercomputing, or high-performance computing power, normally used by military and research facilities, to perform tens of trillions of computations per second, in consumer-oriented applications such as financial portfolios, to deliver

personalized information, to provide data storage or to power large, immersive computer games.

The cloud computing uses networks of large groups of servers typically running low-cost consumer PC technology with specialized connections to spread data-processing chores across them. This shared IT infrastructure contains large pools of systems that are linked together. Often, virtualization techniques are used to maximize the power of cloud computing.[2]

Characteristics and Services Models:

The salient characteristics of cloud computing based on the definitions provided by the National Institute of Standards and Terminology (NIST) are outlined below:

- **On-demand self-service**: A consumer can unilaterally provision computing capabilities, such as server time and network storage, as needed automatically without requiring human interaction with each service's provider.
- **Broad network access**: Capabilities are available over the network and accessed through standard mechanisms that promote use by heterogeneous thin or thick client platforms (e.g., mobile phones, laptops, and PDAs).
- **Resource pooling**: The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand. There is a sense of location-independence in that the customer generally has no control or knowledge over the exact location of the provided resources but may be able to specify location at a higher level of abstraction (e.g., country, state, or data center). Examples of resources include storage, processing, memory, network bandwidth, and virtual machines. [3]
- **Rapid elasticity**: Capabilities can be rapidly and elastically provisioned, in some cases automatically, to quickly scale out and rapidly released to quickly scale in. To the consumer, the capabilities available for provisioning often appear to be unlimited and can be purchased in any quantity at any time.
- **Measured service**: Cloud systems automatically control and optimize resource use by leveraging a metering capability at some level of abstraction appropriate to the type of service (e.g., storage, processing, bandwidth, and active user accounts). Resource usage can be managed, controlled, and reported providing transparency for both the provider and consumer of the utilized service.

Figure 2 : Characteristics of cloud computing

Services Models:

Cloud Computing comprises three different service models, namely Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), and Software-as-a-Service (SaaS).[4] The three service models or layer are completed by an end user layer that encapsulates the end user perspective on cloud services. The model is shown in figure below. If a cloud user accesses services on the infrastructure layer, for instance, she can run her own applications on the resources of a cloud infrastructure and remain responsible for the support, maintenance, and security of these applications herself. If she accesses a service on the application layer, these tasks are normally taken care of by the cloud service provider.

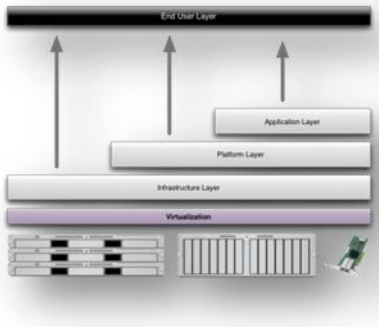

Figure 3: Structure of service models

Benefits of cloud computing:

1. **Achieve economies of scale** – increase volume output or productivity with fewer people. Your cost per unit, project or product plummets.
2. **Reduce spending on technology infrastructure.** Maintain easy access to your information with minimal upfront spending. Pay as you go (weekly, quarterly or yearly), based on demand.
3. **Globalize your workforce on the cheap.** People worldwide can access the cloud, provided they have an Internet connection. [5]
4. **Streamline processes.** Get more work done in less time with less people.
5. **Reduce capital costs.** There's no need to spend big money on hardware, software or licensing fees.
6. **Improve accessibility.** You have access anytime, anywhere, making your life so much easier!
7. **Monitor projects more effectively.** Stay within budget and ahead of completion cycle times.
8. **Less personnel training is needed.** It takes fewer people to do more work on a cloud, with a minimal learning curve on hardware and software issues.
9. **Minimize licensing new software.** Stretch and grow without the need to buy expensive software licenses or programs.
10. **Improve flexibility.** You can change direction without serious "people" or "financial" issues at stake. [6]

Advantages:

1. **Price:** Pay for only the resources used.
2. **Security**: Cloud instances are isolated in the network from other instances for improved security.
3. **Performance:** Instances can be added instantly for improved performance. Clients have access to the total resources of the Cloud's core hardware.
4. **Scalability:** Auto-deploy cloud instances when needed.
5. **Uptime:** Uses multiple servers for maximum redundancies. In case of server failure, instances can be automatically created on another server.
6. **Control:** Able to login from any location. Server snapshot and a software library lets you deploy custom instances.
7. **Traffic:** Deals with spike in traffic with quick deployment of additional instances to handle the load.[7]

LITERATURE SURVEY

This paper describes MediNet, a mobile healthcare system that is being developed to personalize the self-care process for patients with both diabetes and cardiovascular disease. These two diseases were chosen based on their interrelationship. Patients with diabetes are at least twice as likely to have heart disease or a stroke as compared to persons without diabetes. Furthermore, persons with diabetes also tend to develop heart disease or have strokes at an earlier age than other people. MediNet uses a reasoning engine to make recommendations to a patient based on current and previous readings from monitoring devices connected to the patient and on information that is known about the patient. It caters for the uniqueness of each patient by personalizing its recommendations based on individual level characteristics of the patient, as

well as on characteristics that groups of patients tend to share.[8]

Tracking Parkinson's disease (PD) symptom progression often uses the unified Parkinson's disease rating scale (UPDRS) that requires the patient's presence in clinic, and time-consuming physical examinations by trained medical staff. Thus, symptom monitoring is costly and logistically inconvenient for patient and clinical staff alike, also hindering recruitment for future large-scale clinical trials. Here, for the first time, we demonstrate rapid, remote replication of UPDRS assessment with clinically useful accuracy (about 7.5 UPDRS points difference from the clinicians' estimates), using only simple, self-administered, and noninvasive speech tests.[9] We characterize speech with signal processing algorithms, extracting clinically useful features of average PD progression. Subsequently, we select the most parsimonious model with a robust feature selection algorithm, and statistically map the selected subset of features to UPDRS using linear and nonlinear regression techniques that include classical least squares and nonparametric classification and regression trees. We verify our findings on the largest database of PD speech in existence (~6000 recordings from 42 PD patients, recruited to a six-month, multicenter trial). These findings support the feasibility of frequent, remote, and accurate UPDRS tracking. This technology could play a key part in tele-monitoring frameworks that enable large-scale clinical trials into novel PD treatments.

Healthcare information, and to some extent patient management, is progressing toward a wireless digital future. This change is driven partly by a desire to improve the current state of medicine using new technologies, partly by supply-and-demand economics, and partly by the utility of wireless devices. Wired technology can be cumbersome for patient monitoring and can restrict the behavior of the monitored patients, introducing bias or artifacts. However, wireless technologies, while mitigating some of these issues, have introduced new problems such as data dropout and "information overload" for the clinical team. This review provides an overview of current wireless technology used for patient monitoring and disease management. We identify some of the major related issues and describe some existing and possible solutions. In particular, we discuss the rapidly evolving fields of telemedicine and mHealth in the context of increasingly resource-constrained healthcare systems.[10]

This study examined patient and caregiver's perception regarding pervasive healthcare technology using five focus groups and a 31-item questionnaire. To further develop an understanding of the benefits and functionalities that prospective patients deem as either desirable, undesirable, inadequate or in need of further development the study was categorized under 7 main headings: Personal Profile; Benefits; Adoption; Acceptance; Risks; Security, Privacy and Trust; (use of) Cell Phone.[11] This study was completed as part of the European Union BRAVEHEALTH project, aimed at the support of cardiac patients in everyday life using in vivo monitoring and diagnosis, thereby enabling the patient to be more proactive in heath management. Most participants felt that there is a great future for this technology and showed positive response in regards to the potential benefits but are (at present) not willing to adopt the system due to concerns over reliability, like security, privacy and trust.[12]

The U.S. healthcare industry has been given anew mandate to expand the use of health information technology to provide better care and to help reduce costs. Equally, cloud computing is poised to become the fifth utility delivering economies of scale and cost benefits that are difficult for businesses to ignore.[13] The utilization of cloud services for the storage and exchange of personal health information is growing with the use of electronic health records and health information

exchanges. Yet policies and regulatory mandates are still lagging and the potential for the loss of personal information is expanding exponentially.[14] HIPAA/HITECH currently only provides a baseline of protection for personal health information while various IT security frameworks help to standardize the protection and security of personal information as well as the security of cloud services. As the technology matures further and the healthcare industry embraces data and privacy governance programs, the chance for a successful health IT transformation with the use of the cloud significantly increase.[15]

SYSTEM STUDY

FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ECONOMICAL FEASIBILITY
- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

SYSTEM DESIGN

ARCHITECTURE:

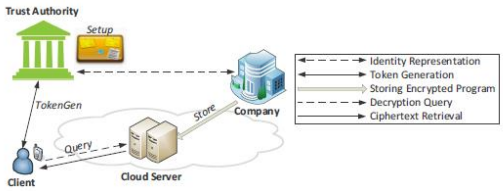

Figure 4 : System Architecture

DATA FLOW DIAGRAM:

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity

that interacts with the system and the information flows in the system.

3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

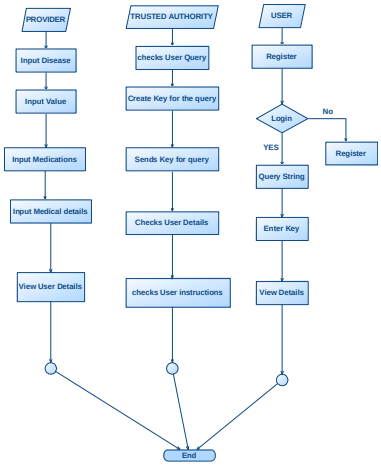

Figure 5 : Data Flow Diagram

UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

USE CASE DIAGRAM:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

International Journal of Research
Available at https://edupediapublications.org/journa
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 03 Issue 04
February 2016

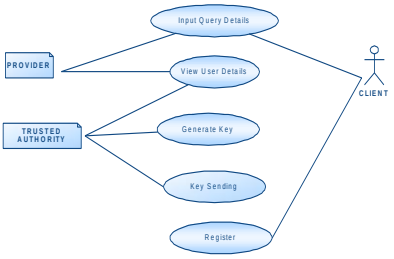

Figure 6 : Use Case Diagram

CLASS DIAGRAM:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

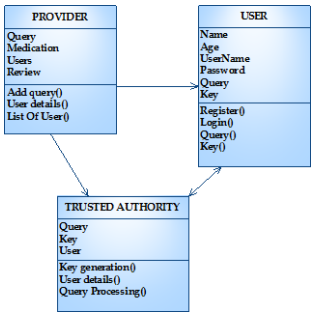

Figure 7 : Class Diagram

SEQUENCE DIAGRAM:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

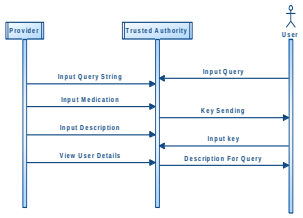

Figure 8 : Sequence Diagram

ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

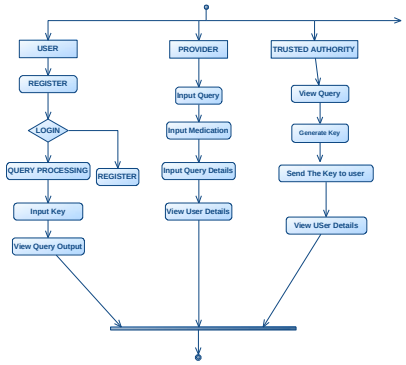

Figure 9: Activity Diagram

INPUT DESIGN

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.
-

OBJECTIVES

1. Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.
2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.
3.When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

OUTPUT DESIGN

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.
1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.
2.Select methods for presenting information.
3.Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

- Convey information about past activities, current status or projections of the
- Future.
- Signal important events, opportunities, problems, or warnings.
- Trigger an action.
- Confirm an action.

SYSTEM ANALYSIS

EXISTING SYSTEM:

Traditional privacy protection mechanisms by simply removing clients' personal identity information (such as names or SSN) or by using anonymization technique fails to serve as an effective way in dealing with privacy of mHealth systems due to the increasing amount and diversity of personal identifiable information.

Traditionally, the privacy issue is tackled with anonymization technique such as *k*-anonymity or *l*-diversity. However, it has been indicated that these techniques might be insufficient to prevent re-identification attack

DISADVANTAGES OF EXISTING SYSTEM:

Unfortunately, although cloud-assisted mHealth monitoring could offer a great opportunity to improve the quality of healthcare services and potentially reduce healthcare costs, there is a stumbling block in making this technology a reality. Without properly addressing the data management in an mHealth system, clients' privacy may be severely breached during the collection, storage, diagnosis, communications and computing.

Another major problem in addressing security and privacy is the computational workload involved with the cryptographic techniques. With the presence of cloud computing facilities, it will be wise to shift intensive computations to cloud servers from resource-constrained mobile devices. However, how to achieve this effectively without

compromising privacy and security become a great challenge, which should be carefully investigated.

PROPOSED SYSTEM:

In this research paper, we design a cloud-assisted mHealth monitoring system (CAM). We first identify the design problems on privacy preservation and then provide our solutions. To ease the understanding, we start with the basic scheme so that we can identify the possible privacy breaches. We then provide an improved scheme by addressing the identified privacy problems. The resulting improved scheme allows the mHealth service provider (the company) to be offline after the setup stage and enables it to deliver its data or programs to the cloud securely.

To reduce clients' decryption complexity, we incorporate the recently proposed outsourcing decryption technique into the underlying multi-dimensional range queries system to shift clients' computational complexity to the cloud without revealing any information on either clients' query input or the decrypted decision to the cloud. To relieve the computational complexity on the company's side, which is proportional to the number of clients, we propose a further improvement, leading to our final scheme. It is based on a new variant of key private proxy re-encryption scheme, in which the company only needs to accomplish encryption once at the setup phase while shifting the rest computational tasks to the cloud without compromising privacy, further reducing the computational and communication burden on clients and the cloud

ADVANTAGES OF PROPOSED SYSTEM:

To protect the clients' privacy, we apply the anonymous Boneh-Franklin identitybased encryption (IBE) in medical diagnostic branching programs.

To reduce the decryption complexity due to the use of IBE, we apply recently proposed decryption outsourcing with privacy protection to shift clients' pairing computation to the cloud server.

To protect mHeath service providers' programs, we expand the branching program tree by using the random permutation and randomize the decision thresholds used at the decision branching nodes.

SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

TYPES OF TESTS

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic

outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot "see" into it. The test provides inputs and responds to outputs without considering how the software works.

Unit Testing:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

IMPLEMENTATION

MODULES:

1. Clients
2. Cloud server
3. Trust authority
4. Company Module

MODULES DESCRIPTION:

First we develop the Clients module, where the client delivers the token for its query to the cloud, which runs the Query phase. The cloud completes the major computationally intensive task for the client's decryption and returns the partially decrypted ciphertext to the client. The client then completes the remaining decryption task after receiving the partially decrypted ciphertext and obtains its decryption result, which corresponds to the decision from the monitoring program on the client's input. The cloud obtains no useful information on either the client's private query input or decryption result after running the Query phase. Here, we distinguish the query input privacy breach in terms of what can be inferred from the computational or communication information. CAM can prevent the cloud from deducing useful information on a client's query input or output corresponding to the received information from the client.

Cloud Server:

In this module CSP has to get the key first. Then only he can store the file in his cloud server. TTP (Trusted Third Party) can only check the cloud server whether the cloud server is authorized cloud server or not. If its fake, TTP won't allow the file to store in cloud server.

Trust Authority:

TA is responsible for distributing Private keys to clients and collecting service fees from clients according to a certain business model such as "pay-peruse" model. TA can be considered as a collaborator or a management agent for a company (or several companies) and thus shares certain level of mutual business interest with the company. In the following, we will briefly introduce the four major steps of CAM: Setup, Store, Token Gen and Query. We only illustrate the functionality of these components here. Because the detailed input and output of those steps might vary in different schemes, we leave more details wherever needed. At the initial phase, TA runs the Setup phase and publishes the system parameters.

Company Module:

The company first characterizes the flow chart of an mHealth monitoring program as a branching program (see Sec. which is encrypted under the respective directed branching tree. Then the company will deliver the resulting Ciphertext and its company index to the cloud, which corresponds to the Store algorithm in the context. When a client wishes to query the cloud for a certain mHealth monitoring program, the i-th client and TA run the TokenGen algorithm. The client sends the company index to TA, and then inputs its private query (which is the attribute vector representing the collected health data) and TA

inputs the master secret to the algorithm. The client obtains the token corresponding to its query input while TA gets no useful information on the individual query.

RESULTS AND CONCLUSION

In this research paper, we design a cloud-assisted privacy preserving mobile health monitoring system, called CAM, which can effectively protect the privacy of clients and the intellectual property of mHealth service providers. To protect the clients' privacy, we apply the anonymous Boneh-Franklin identity based encryption (IBE) in medical diagnostic branching programs. To reduce the decryption complexity due to the use of IBE, we apply recently proposed decryption outsourcing with privacy protection to shift clients' pairing computation to the cloud server. To protect mHeath service providers' programs, we expand the branching program tree by using the random permutation and randomize the decision thresholds used at the decision branching nodes. Finally, to enable resource constrained small companies to participate in mHealth business, our CAM design helps them to shift the computational burden to the cloud by applying newly developed key private proxy re-encryption technique. Our CAM has been shown to achieve the design objective.

REFERENCES

[1] P. Mohan, D. Marin, S. Sultan, and A. Deen, "Medinet: personalizing the self-care process for patients with diabetes and cardiovascular disease using mobile telephony," in *Engineering in Medicine and Biology Society, 2008. EMBS 2008. 30th Annual International Conference of the IEEE*. IEEE, 2008, pp. 755–758.

[2] A. Tsanas, M. Little, P. McSharry, and L. Ramig, "Accurate telemonitoring of parkinson's disease progression by noninvasive speech tests," *Biomedical Engineering, IEEE Transactions on*, vol. 57, no. 4, pp. 884– 893, 2010.

[3] G. Clifford and D. Clifton, "Wireless technology in disease management and medicine," *Annual Review of Medicine*, vol. 63, pp. 479–492, 2012.

[4] L. Ponemon Institute, "Americans' opinions on healthcare privacy, available: http://tinyurl.com/4atsdlj," 2010.

[5] A. V. Dhukaram, C. Baber, L. Elloumi, B.-J. van Beijnum, and P. D. Stefanis, "End-user perception towards pervasive cardiac healthcare services: Benefits, acceptance, adoption, risks, security, privacy and trust," in *PervasiveHealth*, 2011, pp. 478–484.

[6] M. Delgado, "The evolution of health care it: Are current u.s. privacy policies ready for the clouds?" in *SERVICES*, 2011, pp. 371–378.

[7] N. Singer, "When 2+ 2 equals a privacy question," *New York Times*, 2009.

[8] E. B. Fernandez, "Security in data intensive computing systems," in *Handbook of Data Intensive Computing*, 2011, pp. 447–466.

[9] A. Narayanan and V. Shmatikov, "Myths and fallacies of personally identifiable information,"

Communications of the ACM, vol. 53, no. 6, pp. 24–26, 2010.

[10] P. Baldi, R. Baronio, E. D. Cristofaro, P. Gasti, and G. Tsudik, "Countering gattaca: efficient and secure testing of fully-sequenced human genomes," in *ACM Conference on Computer and Communications Security*, 2011, pp. 691–702.

[11] A. Cavoukian, A. Fisher, S. Killen, and D. Hoffman, "Remote home health care technologies:

how to ensure privacy? build it in: Privacy by design," *Identity in the Information Society*, vol. 3, no. 2, pp. 363–378, 2010.

[12] A. Narayanan and V. Shmatikov, "Robust de-anonymization of large sparse datasets," in *Security and Privacy, 2008. SP 2008. IEEE Symposium on*. IEEE, 2008, pp. 111–125.

[13] ——, "De-anonymizing social networks," in *IEEE Symposium on Security and Privacy*. IEEE Computer Society, 2009, pp. 173–187.

[14] I. Neamatullah, M. Douglass, L. Lehman, A. Reisner, M. Villarroel, W. Long, P. Szolovits, G. Moody, R. Mark, and G. Clifford, "Automated de-identification of free-text medical records," *BMC medical informatics and decision making*, vol. 8, no. 1, p. 32, 2008.

[15] S. Al-Fedaghi and A. Al-Azmi, "Experimentation with personal identifiable information," *Intelligent Information Management*, vol. 4, no. 4, pp. 123–133, 2012.