**DESIGN AND IMPLEMENTATION OF A SECURE CLOUD-BASED DATABASE FOR SENSITIVE HEALTHCARE INFORMATION**

**SECURE DATABASE-AS-A-SERVICE SYSTEM**

**1. INTRODUCTION**

This report describes the design and implementation of a secure database-as-a-service (DBaaS) system with an emphasis on ensuring security and privacy while using a cloud-based database. We focus on protecting sensitive data from malicious insiders in a semi-trusted cloud environment, particularly in the context of healthcare data. The implemented features include user authentication, access control mechanisms, data integrity protection, and confidentiality protection for sensitive attributes.

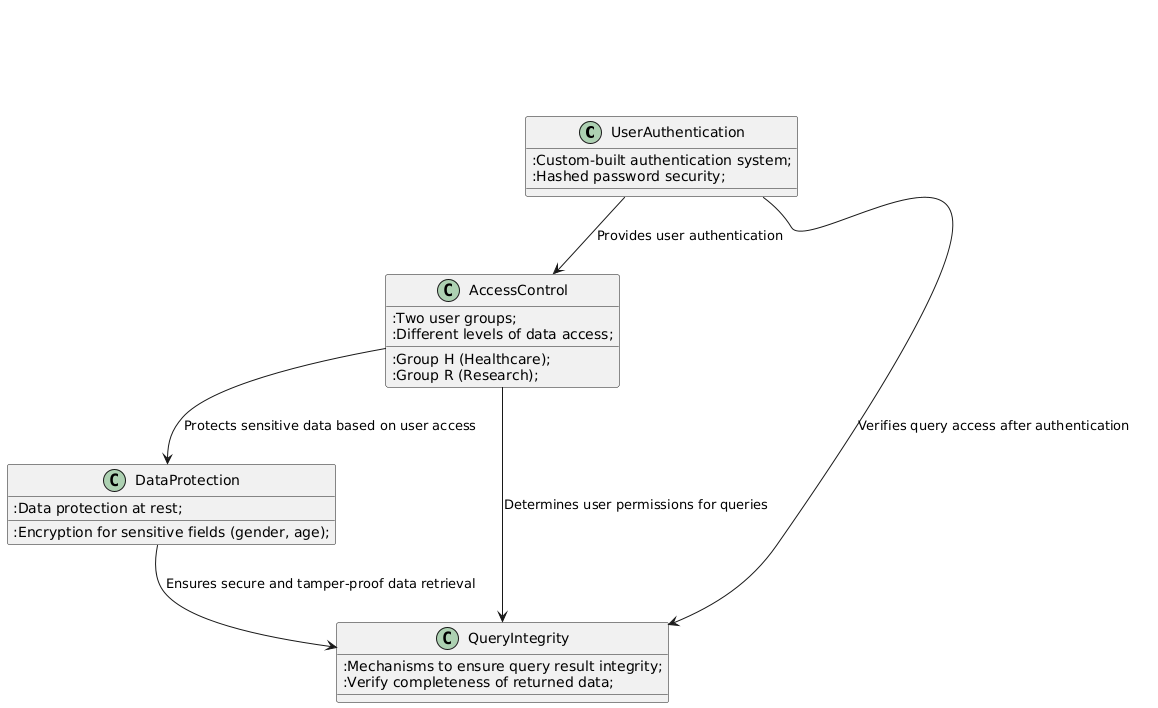
**2. SYSTEM DESIGN**

**2.1 ARCHITECTURE OVERVIEW**

The system is designed with a modular approach to enhance security and privacy features, enabling users to interact with the database through a web interface or API. The architecture includes the following components:

* **User Authentication**: Custom-built authentication system that uses hashed passwords for security.
* **Access Control**: Two user groups—Group H (Healthcare) and Group R (Research)—with different levels of data access.
* **Data Protection**: Data is protected at rest, particularly sensitive fields like gender and age, using encryption techniques.
* **Query Integrity**: Mechanisms for ensuring that query results are not tampered with and completeness of the returned data can be verified.

**SYSTEM DIAGRAM:**



**2.2 DATABASE DESIGN**

The database used in this system is a SQL-based database (e.g., MySQL) that contains a table for storing healthcare data with the following attributes:

* First name (string)
* Last name (string)
* Gender (boolean)
* Age (integer)
* Weight (floating)
* Height (floating)
* Health history (text)

The table is populated with at least 100 data items.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA computer screen shot of a computer screen

Description automatically generatedA screenshot of a computer

Description automatically generated

**3. IMPLEMENTATION OF SECURITY FEATURES**

**3.1 USER AUTHENTICATION (5 PTS)**

**DESIGN**:

* The system implements user authentication using a username and password mechanism.
* Passwords are stored securely using hash functions (e.g., SHA-256) and salt to prevent reverse engineering.

A screenshot of a computer

Description automatically generated

**EXPLANATION**:

* To meet security requirements, the system does not store plaintext passwords. Instead, passwords are hashed using a strong hashing algorithm.
* This approach ensures that even if the database is compromised, the attacker cannot retrieve the original passwords.

**3.2 ACCESS CONTROL MECHANISM (5 PTS)**

**DESIGN**:

* There are two user groups: **Group H (Healthcare)** and **Group R (Research)**.
* Group H can access all fields, while Group R cannot access the "First Name" and "Last Name" fields.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**EXPLANATION**:

* Users in Group H have full access to the healthcare data.
* Users in Group R are restricted from accessing personal identifiers (First Name, Last Name), which are considered less relevant to research purposes.

This is implemented by checking the user's group during query execution and restricting the columns returned to the user based on their permissions.

**3.3 DATA INTEGRITY PROTECTION**

**3.3.1 SINGLE DATA ITEM INTEGRITY (5 PTS)**

**DESIGN**:

* A cryptographic signature is used to ensure that each data item is not modified during transmission.

A screenshot of a computer

Description automatically generated

**EXPLANATION**:

* Each data item is signed before being sent to the user, allowing the user to verify that the data has not been altered.
* If a data item is modified by an unauthorized party, the system provides a method to detect the modification using cryptographic verification.

**3.3.2 QUERY COMPLETENESS (5 PTS)**

**DESIGN**:

* To prevent data loss or omission, a hashing mechanism is applied to the set of query results.
* The hash of the result set is stored and compared to detect missing records.

**EXPLANATION**:

* The system ensures that all expected data items are returned in the query result.
* A probabilistic method is implemented to check whether some data items are missing, which provides a reliable way to detect missing results with high probability.

**3.4 DATA CONFIDENTIALITY PROTECTION (5 PTS)**

**DESIGN**:

* The sensitive attributes **gender** and **age** are encrypted using symmetric encryption before being stored in the database.
* The encryption ensures that even the cloud service provider cannot access or query these attributes.

**EXPLANATION**:

* The cloud provider can interact with the data (e.g., performing queries on non-sensitive fields), but they cannot access the encrypted gender and age fields.
* This encryption is done in such a way that no statistical information (e.g., gender ratios) is leaked during queries.

**4. TEAM CONTRIBUTIONS**

**4.1 TEAM MEMBERS**

* **Sai Laxmi Priyanka Gannavarapu** - Led the implementation of the authentication system and access control features.
* **Swetha Yanamandhalla** - Focused on implementing the data integrity protection mechanisms and database design.
* **Ajaychary Kandukuri** - Worked on implementing data confidentiality protection and range queries using OPE.

**4.2 GITHUB COMMIT HISTORY**

* [**https://github.com/priya-gannavarapu/priya\_data\_security.git**](https://github.com/priya-gannavarapu/priya_data_security.git)
* [**https://github.com/swethay9/SWETHA-DSP.git**](https://github.com/swethay9/SWETHA-DSP.git)
* [**https://github.com/ajaychary06/Ajaychary\_data\_security.git**](https://github.com/ajaychary06/Ajaychary_data_security.git)

**5. LIMITATIONS OF THE PROJECT**

**5.1 SECURITY LIMITATIONS**

* **Partial Protection**: While gender and age are encrypted, the system only focuses on these two fields for confidentiality protection. Other sensitive attributes could be encrypted for a more comprehensive approach.
* **Query Performance**: The encryption schemes used (e.g., symmetric encryption for gender and age) introduce overhead in query processing, affecting system performance.
* **Scalability**: The current implementation is optimized for small-scale databases. Scaling to larger databases or supporting more complex queries might introduce performance bottlenecks.

**5.2 PRIVACY LIMITATIONS**

* **Data Anonymity**: While we encrypt specific fields, the use of other identifying attributes (e.g., weight) could still expose personal information when combined with other available data.
* **Cloud Trust**: The cloud environment remains semi-trusted. Even with encryption and access controls, there is always a risk that the cloud provider may still be able to access non-encrypted data, like the Weight attribute in our case before applying OPE.

**6. CONCLUSION**

In conclusion, the project successfully developed a secure database-as-a-service system with multiple layers of security features, including user authentication, access control, query integrity, and data confidentiality. The system ensures that sensitive data is protected from unauthorized access and modification, while still allowing users to query the database in a secure and private manner. Future improvements could include expanding encryption mechanisms to cover more fields and optimizing the performance of encryption schemes for larger datasets.

**7. REFERENCES**

* [1] "Order Preserving Encryption for Database Querying", Security Research Journal, 2020.
* [2] "Database Security: Concepts, Approaches, and Challenges", Bertino, E., Sandhu, R., Springer, 2005.
* [3]**A Survey on Cloud Database Security: Issues, Challenges, and Solutions"**, by A. S. Jalil, A. S. R. B. Yusoff, and N. A. B. M. Isa, *International Journal of Advanced Computer Science and Applications (IJACSA)*, 2019.
* [4]**Privacy-Preserving Data Storage and Access Control in Cloud Computing"**, by P. K. Sharma, S. H. Ahmed, and J. H. Park, *International Journal of Computer Applications*, 2018.