**IMPLEMENTATION**

Implementation is the stage of the project when the theoretical design is turned out into a working system. Thus it can be considered to be the most critical stage in achieving a successful new system and in giving the user, confidence that the new system will work and be effective.

The implementation stage involves careful planning, investigation of the existing system and it’s constraints on implementation, designing of methods to achieve changeover and evaluation of changeover methods.

**MODULE DESCRIPTION:**

# Number of Modules

After careful analysis the system has been identified to have the following modules:

1. **Authority**
2. **Patient**
3. **Healthcare Service Provider**
4. **Authority Attribute**

# MODULES DESCRIPTION:

# Authority

The principle of least authority states that each component of the system should be given authority to access only the information and resources that it needs for its operation. This principle is fundamental to the secure design of software systems, as it helps to limit an application's attack surface and to isolate vulnerabilities and faults. Unfortunately, current programming languages do not provide adequate help in controlling the authority of application modules, an issue that is particularly acute in the case of entrusted third-party extensions. In this paper, we present a language design that facilitates controlling the authority granted to each application module. The key technical novelty of our approach is that modules are first-class, statically typed capabilities. First-class modules are essentially objects, and so we formalize our module system by translation into an object calculus and prove that the core calculus is type-safe and authority-safe. Unlike prior formalizations, our work defines authority non-transitively, allowing engineers to reason about software designs that use wrappers to provide an attenuated version of a more powerful capability. Our approach allows developers to determine a module's authority by examining the capabilities passed as module arguments when the module is created, or delegated to the module later during execution. The type system facilitates this by identifying which objects provide capabilities to sensitive resources, and by enabling security architects to examine the capabilities passed into and out of a module based only on the module's interface, without needing to examine the module's implementation code. An implementation of the module system and illustrative examples in the Wyvern programming language suggest that our approach can be a practical way to control module authority.

# Patient

This module registers patient details based on general and demographic information. Patients are allocated a Unique Health Identification Number (UHID) and discount cards at the time of registration. Detailed information of patients, Mandatory fields for crucial patient information as per JCI Standards, Alerts in place to prevent erroneous data entry, Generates Smart Card with Unique Health Identification Number (UHID), Advanced multi-criteria search for registered patients, FID wrist band generation and Provision for recording sponsor, insurance and medical tourism details.

This module provides the registration of births and deaths, accurate birth and death certificates, handling of birth and death registers management and generation of statistical reports, updated in real time. Registration of birth data for a newborn, Generation of an ID number when registering a newborn, Registration of family relationships, printing of accurate, secured birth certificates, Providing statistical information and Information can be ported to any Govt. Portals. Registration of death data for a deceased person, printing of accurate and secured death certificates and Information can be ported to any Govt. Portals.

**Healthcare Service Provider:**

The entire health care system is weaved with each other by the single body that is hospital or provider (doctor).

While the other entities include-

* **Insurance company:**Medicare, Medicaid, BCBS, etc.
* **Patient/Consumers:**Patient Enrolled
* **Regulatory Authority:**HIPAA, OASIS assessment, HCFA 1500 and UB92, etc.
* **Health-care and Life-Science solution Vendors**

Basic Terminology of Health Care System

* **Provider**: A health care professional (doctor), medical group, clinic, lab, hospital, etc. licensed by health care services
* **Claim:**A request to your health insurance company to pay a bill for health care service
* **Broker:**An insurance professional, who negotiates, procures insurance on behalf of insured or prospective insured
* **Finance:**Insurance bodies that pay for medical expenses, it could be government (Medicare or Medicaid) or commercial (BCBS)
* **Medicare:** A federal health insurance program for senior citizen and permanently disabled people
* **Medicaid:** A joint and state program that helps low-income families and individuals pay for the cost associated with medical care
* **CPT code**: A current procedural terminology code is a medical code set to describe medical, surgical and diagnostic services
* **HIPAA**: It is a set of rules and regulations which doctors, hospitals, healthcare providers and health plan must follow in order to provide their services

Most health-care organization have adapted software program to process the smooth functioning of the system. This software system gives all the information in a single document for each entity dealing with this. Interconnecting this whole system to a single web application is a huge task and making it work effectively is even a bigger task. Rigorous testing of this health application is compulsory, and it has to go through various testing phases.

**Authority Attribute**

The traditional role-based access control model (RBAC) can not meet the requirements of Service Oriented Architectures (SOA) on the distribution and openness, Attribute-Based Access Control (ABAC), which is more fine-grained in access control, is more fit into the SOA open environment. This paper presents an ABAC-based cross-domain access control system, together with the security domain as a attribute with the subject, object, authority, environment attributes as the basis for access to the decision-making, eliminating integration constraints for the SOA framework based on the RBAC, somehow improves the scalability and alterability of the system, solved the problem of cross-domain access control.

The attribute authority, also known as an attribute store, is a directory or database in which systems can create, read, update and delete (CRUD) consumer attributes. It is a trusted source of consumer attributes to support making attribute-based access control (ABAC) decisions. The attribute authority is considered an identity provider (IdP) that provides user attributes to an attribute consumer. Attributes are encoded into signed tokens, such as Security Assertion Markup Language (SAML) tokens, for consumers to submit to providers to support determination of access privileges.

Cloud computing is a progressive computing worldview, which empowers adaptable, on request, and ease use of Information Technology assets. However, the information transmitted to some cloud servers, and various protection concerns are arising out of it. Different plans given the property-based encryption have been proposed to secure the Cloud Storage. In any case, most work spotlights on the information substance security and the get to control, while less consideration towards the benefit control and the character protection. In this paper, a semi anonymous benefit control conspires Annoy Control to address the information protection, as well as the client character security in existing access control plans. Annoy Control decentralizes the central authority to restrain the character spillage and accordingly accomplishes semi anonymity. Furthermore, it likewise sums up the document get to control to the benefit control, by which advantages of all operations on the cloud information managed in a fine-grained way. Along these lines, display the Annoy Control-F, which ultimately keeps the character spillage and accomplish the full secrecy. Our security assessment demonstrates that both Annoy Control and Annoy Control-F are secure under the decisional bilinear Diffie-Hellman presumption, and our execution assessment shows the attainability of our plans.