National College of Ireland

Blockchain-Integrated Identity and Access Management Framework for Secure IoT Device Management

MSc Research Project
MSc Cybersecurity

Vasu Singh
Student ID: 22243674

School of Computing
National College of Ireland

Supervisor: Prof. Raza Ul Mustafa

National College of Ireland
Project Submission Sheet
School of Computing

National College of Ireland

Student Name:	Vasu Singh
Student ID:	2243674
Programme:	Msc Cybersecurity
Year:	2023-24
Module:	Msc Research Project
Supervisor:	Prof.Raza Ul Mustafa
Submission Due Date:	12/10/2024
Project Title:	Blockchain-Integrated Identity and Access Management Framework for Secure IoT Device Management
Word Count:	5978
Page Count:	19

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Vasu Singh
Date:	16th September 2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Blockchain-Integrated Identity and Access Management Framework for Secure IoT Device Management

Vasu Singh
22243674

Abstract

In the evolving landscape of Internet of Things (IoT) devices, the secure management of non-human entities is important. This research Deals with creating a strong Identity and Access Management (IAM). solution that can be embedded on the IoT devices and based on the blockchain technology, Public Key Infrastructure (PKI), and Keycloak for the improvement. security. The model was deployed on Ethereum blockchain for transaction integrity, data encryption using PKI and Keycloak for authentication and authorization. The study involved a comprehensive evaluation of the framework's performance, focusing on latency, security vulnerabilities and privacy preservation. Results indicated that while the framework effectively managed IoT devices and protected sensitive data and model successfully ensured privacy by limiting blockchain data exposure to non-sensitive metadata, aligning with best practices for privacy preservation.

Keywords- Ethereum, Solidity, PKI, Keycloak, smart contract

1 Introduction

In recent years, non-human entities such as automated systems and deployments of Internet of Things (IoT) devices have become increasingly in demand, transforming various sectors, including healthcare, manufacturing, transportation, and smart cities as a result creating a complex ecosystem where multiple identities and access points interact. There is consistent transition of critical information between these devices and software and protecting sensitive identification and access data in this environment is vital, given the critical role these entities play in processing information. Although automated devices such as sensors etc. and software are trending now days however the security and privacy are still one of the major hurdles in adoption of these systems. As per the research done by Pöhn and Hommel (2023) underscores the importance of ensuring strong security and privacy for automated technologies to prevent data breaches and unauthorized access. Also, non-human accounts are and will be attackers favorite point for gaining access if these non-human entities are not managed like the user accounts Williamson et al. (2022). With advancement of automation, there is a high need to secure identities and attributes in these systems to ensure security within the ecosystem.

The Identity and Access management (IAM) is a fundamental aspect of cybersecurity including the procedures, guidelines and tools utilized to regulate resource access and manage digital identities. In regards with automation, IAM plays a crucial role due to their unique characteristics and system design.

1) Security Challenges: Due to the fact that IoT devices are often applied in a number of extreme conditions, IoT devices are exposed to different types of security threats. Some of these are unauthorized access, data leaks, controlling the device, and the controlling of some of its functions. The inherent structure of Internet of Things is quite distributed and lightweight and therefore may not find the traditional IAM solutions useful since they are designed for the large IT systems Sicari et al. (2015).

2) Privacy Issues: IoT devices are used for collecting a lot of confidential data like environmental data, manufacturing processes, health data and personal data are collected and are sent from the devices. Meeting user's privacy and following the legislation from the territories, such as GDPR, this data should be safe and accessible only to the authorized users Ziegeldorf et al. (2014).

The above research and related studies generate a below research question: How can a secure identity and access management framework be implemented for automated devices, addressing the challenges of securing identities and ensuring privacy?

This research is motivated by need to implement solutions that extend beyond human-centric approaches. By integrating blockchain, PKI and federated identity management and focusing on non-human accounts and zero-trust models, this research seeks to advance the field of IAM, ensuring robust security and privacy for automated devices systems.

Objectives-

- Gather more relevant articles and papers, deep dive into their solutions.
- Review of existing IAM Technologies and their implementation on IoT devices environments, focusing on strengths and weakness of current IAM solutions.
- One of the ongoing challenges is that IAM are failed to preserve privacy which so need to gather real time incident data where privacy was impacted. Along with it, analyze data on security breaches and attacks related to IAM and access control.
- Also, will explore open source IAM frameworks.
- Practical implementation and evaluation of IAM to validate its effectiveness.

2 Related Work

2.1 Blockchain-Based Identity Management in IoT

Kettani and Carnley; 2019 recommend the discovery of IAM solutions that integrate with cloud. Here they also refer about the "Ethereum" which is an open source blockchain platform which does expose it to an independent decentralized global computing infrastructure that helps to integrate a chain of trust into the blockchain technology.

Nuss et al. (2018) present a blockchain-based identity model for such entities Bots, IoT devices, etc. It indicates that identity management is necessary for ensuring that data is protected against an unauthorized actor. That is, the management of devices by their owners through blockchain transactions is realised through this framework. It is noteworthy that most of the existing solutions in Identity and Access Management (IAM) have been tailored to the management of human entities Nuss et al.; 2018and while pointing to the problem of scalability of managing non-human entities' identities, they fail to address this problem effectively. The proposed model addresses this by incorporating

PKI for lightweight encryption and leveraging Keycloak to handle large-scale identity management, allowing for more efficient scaling in IoT environments.

2.2 Blockchain-Based Identity Management for Healthcare IoT Devices

By improving the blockchain-based identity management solution, the work by Alamri et al. (2023) suggests the general framework for healthcare IoT blockchain-based identity solutions where they present a detailed security model concentrating on Health IoT devices, including wearables, in a multi-layered approach because IoT devices and automation have become prevalent nowadays. Here, they defined all the entities present in IoT environment and their inherent weaknesses which include no physical security and no authentication and authorization, the threats associated with these weaknesses which include device tracking, signal jamming and battery drain attacks necessary for the formation of a strong security architecture. Nevertheless, the research by Alamri et al. (2023) doesn't recognize a particular type of IoT devices since each gadget comes with its problems, and it fails to address the issues of Health IoT systems or devices / software that is used a part of daily life such as wearable intelligent devices or any sensor that is applied in supply chain businesses. Also, the proposed solution has not been practically implemented and not been evaluated by experts as this technology is new in the security domain. Practical implementation and evaluation are crucial step for validating new security technology.

2.3 IoT Identity Management Using Distributed Ledger Technology

The proposed system by Sadique et al. (2023) involves an identity management model for edge IoT devices, DLT, and smart contracts. The model is founded on the use of both private permissioned distributed ledgers and private consortium ledgers since identification details need to be protected. The material is geo-situated, its general, extensible and includes strict consensus procedures. It employs a two-tier structure for its layers namely- Fog layers and Cloud layers. Both devices retain their identity as well as the identity of the connected gateway through the usage of secure public and private key communication. credit information are exchanged beween local identity providers and Global identity providers without revealing the private key thus enhance security. This goes a long way to stress the great approach in strengthening the security and privacy in IoT identity management through decentralization and usage of sophisticated cryptographic methods. While this seem logical, but it was only done on experiments the practical solution has not been implemented in real life problems. This raises questions about its practicality as well as its applicability to large organisations. Secondly In DLT technologies, transparency is inherently built which does trust but restricts the privacy hence the sensitive information can be easily leaked to an insider threat.

2.4 ERC-20 Smart Contract

This paper by Priest et al. (2023) explains the process of how to compile and launch an ERC-20 smart contract on Ethereum, while at the same time stressing the long-term

stability as a key factor for the development of cryptocurrencies. Here the authors successfully discuss both the topic of smart contract development and the context of its impact on the earth's environment thereby contributing to the discussion of matters related to environmentally friendly blockchain platforms. Nevertheless, despite the fact that the paper is somewhat informative in giving a background about sustainable practices, it is rather weak in outlining the methodologies for implementing sustainable practices' measures within the ERC-20 systems. Discussions aimed at reducing the energy consumption rate of blockchains are general with no common guidelines or benchmark that could be applied across various blockchain projects with the intention of containing the carbon footprint. In addition, the paper lacks appropriate depth of the analysis of potential economic effects arising from the use of sustainability features in smart contracts. Altogether, the work is a good reference point to start the contamination around the theme of sustainable blockchain technology but it lack more specific and practical pedestrian suggestion. The proposed model adopts Ethereum's Proof-of-Stake mechanism, which reduces energy consumption compared to Proof-of-Work, thereby offering a more sustainable solution suitable for resource-constrained IoT systems.

2.5 Blockchain-Based IoT Security Models

In the work of Spadavecchia et al. (2024), some new ideas on the action of using blockchain to deliver secure data in IoT, especially for using Hyperledger Fabric and Ethereum in the multi-layer architecture. From the provided directions of the authors, the authors are able to present how this architecture can be implemented practically using a Raspberry Pi, argue the applicability of the blockchain solutions in the smart city contexts. In this case, while the paper presents a solid foundation, it is weak in exemplifications and actually does not go deep into analyzing the feasibility of applying the proposed solution on a large scale, especially in an environment with a great number of IoT devices. Furthermore, the use of both public and private blockchains, which is rather original, also has certain concerns as to the efficiency of such a model for the future and the complexity of its management. This research is somewhat connected to my proposed model since it focuses on the use of a blockchain-based model for safe transmission of data in IoT networks. The proposed model builds on this by focusing solely on Ethereum's public blockchain, simplifying management while maintaining security. The actual experiences obtained from its application can be used to improve our model concerning the interaction of multiple layers of the blockchain system in order to maximize their security and performance.

2.6 Blockchain Solutions with Secure PKI Systems

The research by Singla and Bertino (2018) analyses the problems associated with the mainstream CA-based (PKI) system and suggests the three different blockchain-oriented solutions with the help of the Emercoin NVS technique, the use of the Ethereum-based smart contracts, and the Light mode synchronization in Ethereum. The authors have also compared all these blockchain approaches appropriately that these approaches are better in term of computational complexity, storage complexity, and security as compared to the PKI approaches. This performance occurs due to the right benchmarking with IoT devices and a real-world implementation of the paper's algorithms, which indicates blockchain's utility for scalable and secure PKI systems. Nevertheless, there is a possibility that

the authors should have covered more extended investigation into the career direction for scalability and the threats of security of the blockchain with the rising size of the network, especially with IoT restricted resources. It correlates with the application of Ethereum in building secure decentralised applications for IoT and understand how smart contracts and block chain technology can improve the security and functionality of IoT. The proposed model integrates PKI with blockchain to handle device authentication while addressing scalability concerns through Keycloak's federated identity management system and makinf sure that solution should remain efficient in case there are high volume of devices.

2.7 Challenges in Ethereum

Blockchain network has its own barriers, especially time to scalability and for security, Ethereum network suffered a lot. Chen et al. (2020)accurately presented a list of security concerns pertaining to Ethereum and the possible areas that may be attacked within the Ethereum network. His analysis highlights the importance of security mechanisms to secure the integrity of contracts as well as the blockchain network. The security issues are directly addressed in the proposed model through security audits of the smart contracts using tools like Slither.

2.8 Comarison of Ethereum and Hyperledger

Zhao (2022)in his paper done detailed comparison of available both open source blockchain. The blow table summarize the comparison done.

Parameters	**Ethereum**	**Hyperledger**
Governance	Managed by Ethereum developers (open-source community)	Managed by the Linux Foundation
Type	Public, Permissionless	Private, Permissioned
Smart Contracts	Uses Solidity for writing smart contracts	Uses Chaincode (supports multiple languages like Go, Java)
Use Case	Mostly for Peer-to-Peer (P2P) and Business-to-Consumer (B2C)	Mostly for Business-to-Business (B2B) operations
Privacy and Confidentiality	Public data is accessible to all participants	Supports private and confidential transactions through channels

Table 1: Comparison of Ethereum and Hyperledger

Ethereum blockchain is relatively decentralized and has no restrictions on the users. Therefore, suiting my research for the application of the decentralized applications. Automated and trustless transactions are vital to provide a secure system and its smart contracts written in Solidity are crucial for it. Sustainability is applied to blockchain operations, altering the mechanism of choosing the block creator from Proof of Work to Proof of Stake in Ethereum. Although Ethereum has certain issues with performance and scalability, which have been already mentioned by Chen et al. (2020), the general

acceptance of Ethereum and its well-developed ecosystem offer necessary prerequisites for the implementation of decentralized solutions, which makes Ethereum an appropriate solution for my research.

The overall analysis of the topics under review shows how identity management for blockchain-enabled IoT devices is advancing. Existing frameworks offer a good point of origin however they have issues with scalability, privacy and actual deployment. This proposed solution fills in these gaps by utilising PKI for strong encryption and authentication, Keycloak for scalable identity management and Ethereum's decentralised blockchain for secure and immutable identity management. The model overcomes the theoretical limits found in earlier publications by demonstrating its applicability in real-world IoT scenarios through practical implementation and evaluation.

3 Methodology

The focus of this research is on the deployment of Ethereum blockchain technology by incorporating Smart Contracts, Keycloak for Identity Management and Public Key Infrastructure (PKI) to implement an effective and efficient IoT device management system that has privacy constraints. The research employed a systematic approach of which it followed several steps that include literature review, design of the model, implementation of the model and finally evaluation of the proposed model. In this section, the research procedure followed will be described, tools and technologies that were used and an evaluation of the model will be made.

The study was initiated by identifying the current IAM technologies and the ways in which they are implemented in the IoT and automated settings. The literature review also underlined what are the proposed IAM solutions in the current literature and their strengths and weaknesses concerning scalability, privacy, and security in decentralised contexts. Previous works by Kettani and Carnley (2019), Nuss et al. (2018) have defined ideas that our works embraced as a backbone of the developed IAM framework. The framework was developed using various tools and technologies: OpenSSL for PKI, Keycloak for authentication management and Ethereum with solidity smart contracts for secure transaction management. The selection of these tools was done with the view of their effectiveness, reliability and suitability in achieving the intended research goals as earlier other studies had utilised.Adams and Lloyd (2003).

One of the critical design choices in this research was the selection of Ethereum as the blockchain platform for managing secure transactions. Ethereum is one of the most mature blockchain platforms makes it a reliable choice for integrating decentralized applications and smart contracts. In addition to that, Ethereum has a well-established security framework and many vulnerabilities have been identified and addressed over the years, making it one of the most secure public blockchains available. Also, to guarantee that data transferred between IoT devices and the blockchain is encrypted and authorised, PKI was integrated into the design. PKI is widely utilised in many secure systems and provides an effective method for secure communication. The choice of Keycloak as the Identity and Access Management (IAM) solution for this framework was based on its capacity to manage large-scale users' and devices' identities and access. It was considered the best option for handling user identification and authorisation in a decentralised sys-

tem because of its support for federated identity, role-based access control (RBAC), and interoperability with contemporary authentication protocols like OAuth2 and OpenID Connect.

The experiments were targeted to the main features of the IAM framework that contains latency assessment, security audit, and privacy-preserving analysis. The collection of specific data was done through the performance of certain CLI commands to get the time required to create as well as invoke IoT devices from the Ethereum blockchain. The time taken by these operations was noted down with the help of time command in the shell of the MacOS and the split up of the user time, system time and total time was carried out to see the reasons for the time lag observed. Security assessments were conducted using Slither, a Solidity static analysis tool that would help determine the various loopholes that may have been made in the smart contracts. The identified problems included wrong use of modifiers or the presence of outdated versions of Solidity that may cause security vulnerabilities Chen et al. (2020); Sadique et al. (2023). Comparison was also made between the creation of devices and the retrieval so as to find out areas that have poor performance. The aspect of privacy was assessed from the metadata generated on the blockchain during transactions without exposing the PII. Christidis and Devetsikiotis (2016); Singla and Bertino (2018).

The given approach was carefully planned to avoid missing any aspect of IAM framework's efficiency in protecting IoT device management. In line with that, the research applied systematic use of Technologies and Tools to maintain reliability and validity of the outcomes.

4 Design Specification

The main objective of this research is to propose a model that integrate IoT systems with Block chain technology with PKI for authentication and Keycloak for access control mechanism to build up a secure and privacy preventing device management model.

4.1 Core Components

Ethereum Blockchain: In the case of IoT devices, Ethereum which is an open source blockchain Kettani and Carnley (2019) utilizes blockchain as a decentralised record of transactions so that the recording of transactions is credible, transparent and immutable. This is important in IoT specifically because the devices involved are expected to transmit data and self-organize. As discussed by Wood et al. (2014), it runs smart contract that enable automated registration of devices, data storage and access management. The below figure will provide blockchain architecture and actual model blochain blocks.

Smart contracts: Help in the execution of the complex processes recognised to be involved in the management of Internet of Things devices. Information acquired by these contracts keeps user and device rights, verifies device data and logs Ethereum blockchain transactions. Smart contracts enable security and privacy in the performance of activities, such as device registration, developed under Solidity Christidis and Devetsikiotis (2016). Smart Contracts contain access control policies based on the permissions in the JWT token which results from the authenticated user's identity and any effort to gain access with a different token is forbidden.

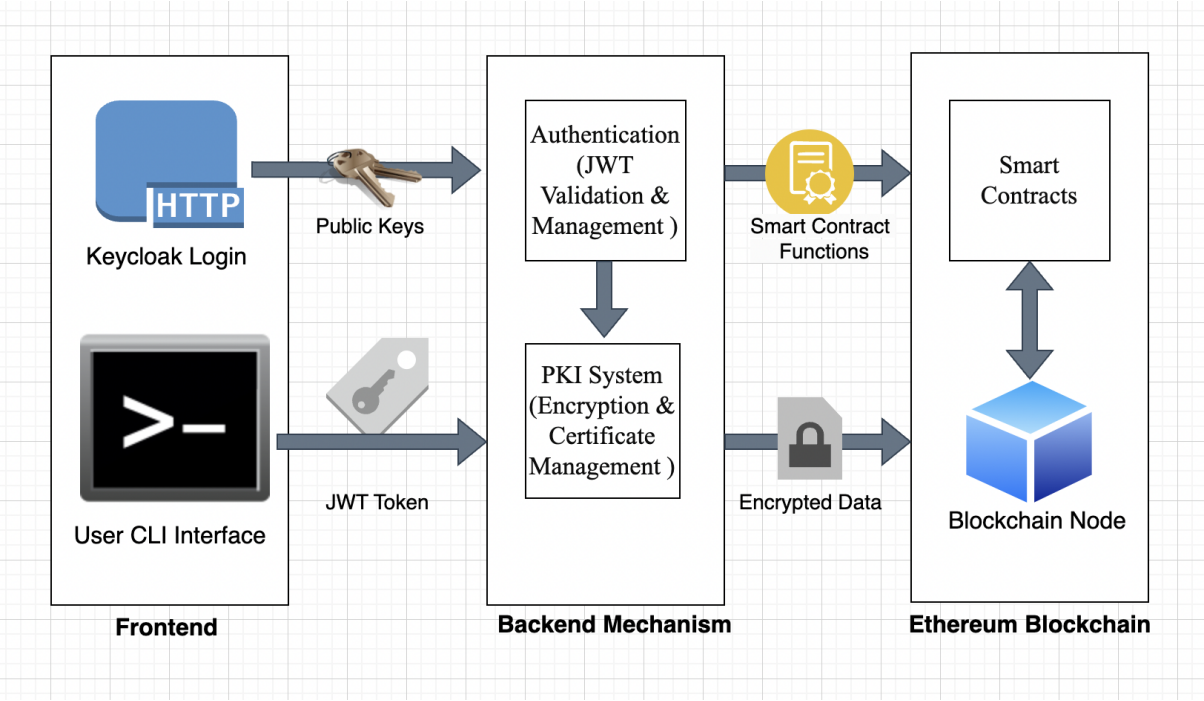

Figure 1: Proposed Model Design

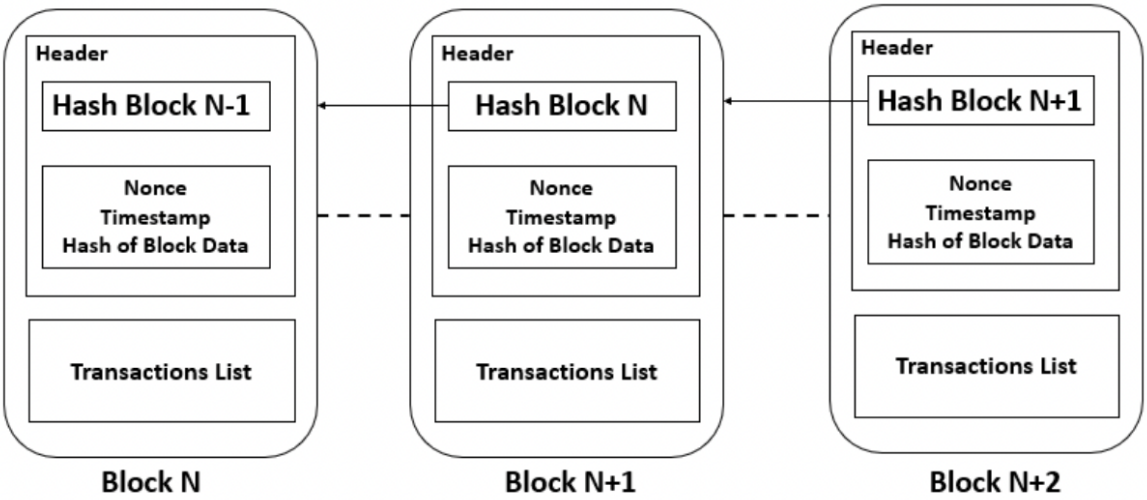

Figure 2: Blockchain architecture BOURIAN et al. (2023)

Figure 3: Model Blockchain Blocks

Public Key Infrastructure (PKI): As discussed by Adams and Lloyd (2003) PKI solutions are critical for safeguarding the data breaches and unauthorized access specifically for the systems dealing with important IoT data. It safeguards the communications occurring between devices and the Blockchain Network. Oversees digital certificates which are utilized to encrypt the data transmission and also assure the legitimacy of device identities in order to safeguard the confidentiality and integrity of the data.

KeyCloak: In the same manner that Keycloak regulates user, device and backend services interactions in more secure manner, Internet of Things systems can incorporate it to achieve a similar objective. It can safely connect the IoT devices by creating a new realm account with Keycloak. Each of the devices has the client identification and secret, these can be considered as the further client identification. It will be possible through Jason Web Token (JWT), which Keycloak issues and that devices use to authenticating requests to backend services Chatterjee and Prinz (2022). It ensures that only those devices that are permitted can access the available system resources. Tokens that are used in IoT devices are typically of a short duration and need to be replaced occasionally to ensure that there is no potential threat to security.

4.2 Backend Processs

As a middleware, backend implements the layer between the front end (CLI) and the back end comprising of blockchain and IoT devices. It translates and process CLI calls, ensuring that proper action is performed on the blockchain and data is properly addressed. Let's have a deep discussion how backend handling and interacting with each components of model.

Keycloak Integration: Authorization and authentication in the backend of the project is managed by Keycloak. If for instance, specific device sends a request to Keycloak, it then validates the credentials using the database. It ensures that all the backend services are accessible only by the verified and rightful users or only by the devices that have been approved. After a user has been authenticated, Keycloak provides tokens (JWTs) which the backend uses to subsequently authenticate the request. This token assists in the backend to enforce the security policies since it contains information about the role and the rights of the user.

PKI: PKI is also utilized by the backend to handle the storage and security of data in and leading to the foundation of ensuring safe transmission. Public keys are used to encrypt data from the IoT devices or the user interface (CLI). Backend utilize matching private key to decrypt the processed and/or stored data before actual processing or storage. The backend also ensures that the data in its communication with IoT units, the backend, as well as Blockchain, are encrypted throughout the process to prevent unauthorized access or interference.

Smart Contracts and Ethereum Blockchain: The backend generates the required transaction, signature with the corresponding private key and pushes it to Ethereum network when an event occurring and initiates a blockchain operation. Thereafter, it waits for acknowledgement from the template before it sends back the result to the CLI.

4.3 Model Overview

This models are an extended model with the ability to protect IoT devices and interact with them since they incorporate several technologies. Fundamentally it uses Ethereum blockchain which is a backbone of this model to make sure that any interaction with IoT devices has their activities noted in such a manner that cannot be manipulated or altered. This is accompanied by smart contracts, which are self-executing programs that perform such functions like registering of devices and regulating the access to them based on the predefined parameters such as only allowed users can register or handle devices etc. Also, using PKI the retriever system incorporates parts of protecting data and identification of devices and users where the information is encrypted and can be decrypted only by use of the correct keys. Keycloak is being employed for the roles and permissions, which defines who can do what in the system. It is an identification and authentication tool that uses secure tokens to verify the user and his or her rights. Altogether, these components will build a strong security system to restrict the unauthenticated individuals' access to IoT devices in addition to ensuring the security and immutability of any interaction with those devices. This makes it a good and secure IAM framework that is specifically designed to address the needs of entities, especially IoT devices.

4.4 Workflow of Model

Let's discuss the step by step workflow of the model-

User Authentication: The user start the login process through CLI, there user have to provide their Keycloak username and passwords. Then CLI will authenticate user through Keycloak server from a specific endpoint. After successful authentication the Keycloak server issue a JWT token which returned as a output over CLI.

Device Registration: The authenticated user request to register a device through CLI with details such as device ID, its type, public key and JWT token and then CLI sends this request to backed middleware.

Token Validation: The backend validates the provided JWT token using Keycloak's public key to make sure that request is made from valid user. After successful validation it the backend encrypts all the details using user's public key (which was provided at time of registering the device).

Blockchian Transaction: These encrypted package sent to Ethereum Blockchain in the form of transaction. Then smart contract receives the encrypted package and register device. And upon successful registration it display message on CLI that **"Device successfully created"**.

Device Data Retrieval: The user submit request through CLI to get the device information, where it asks for JWT token which contains user's identity. This JWT token again validated by backend to check user's information. After this, backend sends query transaction to blockchain which further processed by smart contracts and retrieves encrypted device data using device ID provided by user.

Data Decryption: After successful data retrieval, the backend decrypts the data using private key obtain from PKI system. Then decrypted details displayed over CLI to user.

5 Implementation

The implementation part is focused on creating secure Identity and Access management framework for IoT device management integrating with various technologies. It incorporated with Ethereum blockchain which in open source as discussed by Kettani and Carnley (2019), along with Public Key Infrastructure for encryption and Keycloak for authorization ad authentication. The interaction with model takes place through CLI for making operations smooth and maintaining security of core components. Let's discuss the implementation of each part with tools and technologies used.

5.1 PKI Installation

For the purpose of providing and maintaining a secured method of communication and data transfer between the various components of the system by handling the digital certificates and encryption keys. OpenSSL[1] was installed to create and control the PKI system, including the public and private keys as well as digital certificates. The PKI system adopted is used for management of certificates for both the users and the Iot devices which it issues. Digital certificates were created for each IoT device and the users, this helped in encryption of data flow as well as ensuring that even the devices and users accessed the system in a correct manner.

5.2 Keyclaok Setup and Configuration

As for the user authentication and the restriction of the access to the IoT devices and the blockchain, the user management module should be designed to allow interaction only with the authenticated users and provide the role-based access to the system. Keycloak, an open source identity and access management as discussed by Chatterjee and Prinz (2022) was used in the project to control authentication, sessions and to generate Jason Web Tokens (JWT) for API access.

Realm Creation : A new realm, called IoTrealm, was developed in Keycloak and it is the specific environment for managing user identities and their rights in the IoT system only.

Client Configuration: Within the IoTrealm, a client called Blockchain-client was developed with the access type of confidential to help with the communication between the CLI and the Keycloak server securely. It is also configured to accept authentication requests from the CLI as will be seen later in this paper.

Token Endpoints setup: Token generation and certificates download endpoints were configured, thus the CLI was capable of authenticating users and obtain imperative cryptographic keys from Keycloak.

Token Endpoint: http:*//localhost:8080/realms/IoTrealm/protocol/openid-connect/token*
Certificate Endpoint: *http://localhost:8080/realms/IoTrealm/protocol/openid-connect/certs*

Therefore, an operational Keycloak realm backed with a client meant for securing users in their interaction with the system using the CLI and for the issuance of JWT tokens that granted authenticated and authorized access to the blockchain.

[1]https://www.openssl.org/

5.3 Smart Contract Deployment

For the efficient handling of the IoT devices on the blockchain as well as the processes of the IoT device registration, storing and access. Smart contracts also work to guarantee that all the process that is carried out is secure and has enough transparency. Solidity [2] version 0.8.13 was used to write the smart contracts and Truffle, a development framework for Ethereum, was employed to compile, test and deploy these contracts to the Ethereum network. Ganache was used as a local blockchain for development and testing.

Smart contacts compilation: Smart contracts were coded in Solidity to perform IoT device registration and record management as well as entry authority. These contracts made it possible to make all the interactions to be secure, permanent and traced. The contracts were compiled through Truffle so as to confirm that the structures would be correct and there were no typo mistakes before deployment.

Migrations scripts: Scripts were created specifically to use the in the Ethereum blockchain for the deployment of smart contracts. These scripts also oversaw the process of deployment and where contracts are to be deployed on the blockchain.

Smart Contract Deployment: The contracting systems that can autonomously execute the contractual terms the smart contracts were deployed to the local Ethereum using the Ganache. This deployment was done through Truffle, migration scripts are used to manage the sequence of deployment

Complied and deployed smart contracts on the Ethereum blockchain in the overall successful manner. They regulate the functioning of IoT devices and guarantee the data's consistency and protection in the blockchain. Deploying of smart contracts became easier by using migration scripts and minimizing on the chances of getting errors.

5.4 Data Encryption and Storage

Before storing it on the blockchain, device data was secured with the help of the PKI system. This process guaranteed that so much information would not be easily visible, even if the actual blockchain was open to public viewing. Using the smart contracts, encrypted data was saved in the Ethereum blockchain. That is why, the data itself was stored in the blockchain with references (in the form of hashes) to it.

5.5 Command Line Interface (CLI)

The CLI uses Web3.js [3] to communicate with the deployed smart contracts on the Ethereum blockchain. The FTC core performs tasks such as; sending transactions, getting data, and handling device details in an efficient and safe manner. The CLI was linked to Keycloak to perform user authentication with the help of JWT tokens. Thus, the integration rendered only the legitimate user capable of performing the commands.

A fully functional CLI which enables the user to control IoT devices and ensure the security of the system. The CLI provides guarantees that all the performed actions are identified and approved, all transactions with the system being recorded on the blockchain. Customers can register new devices, perform read operations to get data of the devices, and even manipulate the state of the devices from the CLI.

[2] https://soliditylang.org/
[3] https://web3js.org/

6 Evaluation

6.1 Model Latency

This testing is conducted with the intent of determining the latency that is incurred by CLI implementation of creating and retrieving IoT devices. The latency of the system will be calculated in terms of time taken to create one of the devices and to fetch some of the data. Here "time" command is used to capture the time which takes to run the script with the given functions.

6.1.1 Device Creation/Registration

Following command was executed to create device-

Command: *time node cli.js* with **create** function and further provided necessary device details such as device ID, type with valid tokens and certificates which are required for the device registration.

The duration of the command to complete was ascertained by making use of the shell's timing facilities.

Output: *node cli.js 0.53s user 0.07s system 1 cpu 54.431 total*

Time recorded: user time 0.53sec

System time 0.07sec

Total- 54.431 seconds

Actual CPU time used in user and system mode is much lesser than total time which means that most of the time is spent in waiting either for valid tokens, block chain confirmation etc.

6.1.2 Device data Retrieval

Following command was executed to get device information-

Command: *time node cli.js* with **get** function where we provided device ID and only valid JWT token.

Output: *node cli.js 0.41s user 0.05s system 2 cpu 16.581 total*

Time recorded: user time 0.41sec

System time 0.05sec

Total- 16.581 seconds

That a total time is lower than the time of the device creation step can be explained by the fact that retrieval operations are shorter, presumably, because there is no confirmation of blockchain transactions, which always take more time.

6.2 Smart Contract Security Audit

The goal of this experiment is to evaluate smart contract with the help of Slither[4], an open-source Solidity static analysis tool. The purpose is to find out security risks, bad or suboptimal code, and deviation from standard conventions in the Solidity language.

The analysis is done by running the command slither *contracts/¡solidity filename¿* in the command line interface. The tool is able to scan for different problems like the wrong application of modifiers, prescripted versions of Solidity and error in declaring variables. *contracts/¡solidity filename¿*

[4]https://pypi.org/project/slither-analyzer/0.8.3/

Output: The technical analysis performed using the Slither tool found out that the restricted modifier in the **migration.sol** contract does not always execute the function body or revert a transaction. This can lead to unpredictable behavior in the smart contract, meaning that a function may not do what is required of it when certain conditions are met.
This vulnerability has a significant impact since it puts the system's integrity at risk by allowing unauthorised operations or incomplete transactions. Due to the restricted modifier unpredictable behaviour, transactions may not be properly validated, which leaves the system vulnerable to attacks like unauthorised access or function abuse like false device registration.

Furthermore, as for the Solidity language used, both **migration. sol** and a **IoT-Device. sol** the version of Solidity contract **(=0. 8. 13**) is known to have several critical problems, for example, memory management, selector access and changes in the location of data. These issues can lead to vulnerabilities or unexpected behaviour in smart contracts but while working with its latest version 0.8.20, it was leading to compilation error. So, after thorough analysis on open forums it is found that moving to version 0.8.13 will resolve compilation issue.

6.3 JWT Token Validation

The main reason why this is required is to assess the performance of JWT (JSON Web Token) validation via the CLI by coming up with different varieties of tokens, valid and invalid, altered and old, and see how the CLI will utilize them. This will be useful to enhance our capability in distinguishing token states to the system.

Here, generated a JWT token using below command by providing required details-

curl –data "client-id=" –data "client-secret=" –data "username=" –data "password=" –data "grant-type=password" "http://localhost:8080/realms/IoTrealm/protocol/openid-connect/token"

As a result, it generates JWT token along with expiration time and refresh token which can be used to re-generate the JWT token again using that.

6.3.1 Valid Token

The generated JWT token was provided to while executing *cli.js* script with create device function along with all required details.

Output: Successfully able to create/register device with message display **"Device created successfully"**

6.3.2 Tampered Token

This time the same JWT token was altered by changing few bits of it, and executed the same process as above.

Output: Getting error message that **"Inavild Token"** and unable to create device.

6.3.3 Expired Token

As mentioned above, the JWT token generated with expiration time, token was provided after set expiration time while running the script.

Output: Getting error message that **"Inavild Token"** and unable to create device.

Valid, tampered and expired token are used to test the capability of the system to offer appropriate authentications for the JWT tokens to restrict certain actions. This is important in terms of security and restricting unauthorized people from accessing such an application whenever JWT is used for the authentication of an application.

6.4 Privacy Preserving Evaluation

The privacy-preserving capabilities of the proposed model using the Ethereum blockchain, particularly focusing on the visibility of transaction details and the protection of Personally Identifiable Information (PII). Whenever a new device is registered or created into the system after executing the command, the transaction was recorded on the Ethereum blockchain.
Outcome: Only following transaction details were observed on the blockchain which can be observed from the below figure:

Transaction Hash
Gas usage
Block number
Timestamp

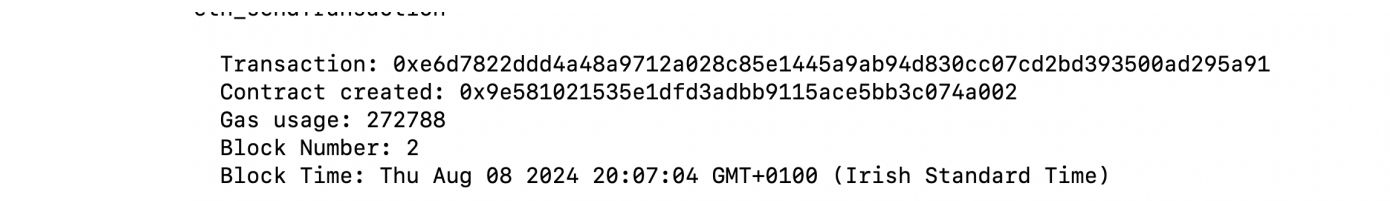

Figure 4: Blockchain Transactions

The actual data to be recorded on the blockchain are only the transaction inputs and outputs, but the data stored on the block includes only certain essential details like transaction hash, gas used, block number, and block time. While sharing data on the device or using it, no specific facts like a person's identity or PII information is written on the blockchain. While blockchain ensures virtually all the participants in the network have equal visibility into transactions, and the implementation of the smart contract does occur on a public ledger, some preconditions ensure that no detrimental information can be released.

6.5 Discussion

The assessment of different aspects of the proposed model and how they can be strengthened, the overall assessment of the effects of the proposed solution in the context of managing IoT devices as facilitated by Ethereum blockchain, Keycloak, and PKI. The latency analysis shown: Although the model is fast in device discovery phase ever creating involves certain delay which is mainly due to time being taken in receiving acknowledgement from the blockchain network. This imply that while the model leverages on the security of blockchain system and the immutability it has inconveniences of blockchain system

performance. The security audit of smart contracts revealed that there are some risks, especially connected with the non-uniform application of the restricted modifier during the migration. sol contract. Moreover, having chosen an older Solidity version, the product became vulnerable to known vulnerabilities. Earlier, the latest version of Solidity was used; however, due to the compilation error problem prevalent in the new version continuing with the previous version is a feasible solution.
The model was able to perform well in the validation of JWT token, and the system was able to differentiate between the three types of tokens which included; valid, tampered and expired tokens. This capability is rather important for the purpose of sustaining the access control in the unconventional environment of DEP. The privacy preserving features of the model were verified with the help of an analysis of the data on the blockchain. The results included the following; one could only observe other non-identifying details such as the transaction hash, gas consumption, and block numbers and no PII. In this regard, this approach fits well with privacy best practice so that even though blockchain is revolutionary, the transparency that comes with it never infringes on the privacy of the users.

7 Conclusion and Future Work

The aim of this work was to devise and assess the first stable IAM paradigm for non-human identification, that is IoT devices, which is to include blockchain, PKI, and Keycloak IAM systems. This framework was intended to solve such problems as identity protection and, in general, privacy in the world where new entities such as IoT devices and Which will be important. Thus, this research was able to develop and test a model that incorporates the above mentioned technologies. The model showed good signs of achieving efficient protection of IoT devices in terms of the identities themselves and the data associated with those identities. The model delivered device registration and device retrieval commands utilizing a command line interface (. CLI). But the measurements of latency showed that although device deletion or retrieval took a very short amount of time, device creation took a lot longer because of the time involved in completing the blockchain transaction. Security checkpoints analysis with the help of the tool Slither revealed certain risks, such as those connected to modifiers and used Solidity version. Though it was necessary to use an older version of Solidity to prevent the issues with the compilation, this situation has drawn attention to the fact of the existence of the risks related to the usage of the outdated software. Further, the proposed model was less transparent in the sense that only transaction data was stored on the blockchain while other details were kept anonymous. Meta-information is only logged, which is not sensitive, for instance, data including the hash of the transaction and the amount of gas used; no Personal Identifiable Information (PII) is saved. It is in concordance with the recommended measures of personal privacy which further supports the credibility of the model in the expectation to safeguard the users' information.

Future Works: It is clear from the research that blockchain, PKI, and Keycloak can be effectively utilised for the identity management of IoT devices and provide privacy. Nevertheless there are some drawbacks revealed within the use of this approach including the delay in the creation of device type contracts and potential security weaknesses connected to utilization of less variant Solidity versions. In addition to that, the model

doesn't have a proper user interface as all the interactions were performed using CLI, a proper integration of UI can be considered for further advancements. Also, this model is implemented on a single personal computer and with simulation environment. So, the implementation and evaluation of model with added layer of security mechanisms must be conducted in real world enterprise environment with actual IoT devices as real world implementation has their own set of unique challenges.

References

Adams, C. and Lloyd, S. (2003). *Understanding PKI: concepts, standards, and deployment considerations*, Addison-Wesley Professional.

Alamri, B., Crowley, K. and Richardson, I. (2023). Cybersecurity risk management framework for blockchain identity management systems in health iot, *Sensors* **23**(1): 218.
URL: *https://doi.org/10.3390/s23010218*

BOURIAN, I., SEBBAR, A., CHOUGDALI, K. and Amhoud, E. M. (2023). Sshceth: Secure smart home communications based on ethereum blockchain and smart contract, *GLOBECOM 2023 - 2023 IEEE Global Communications Conference*, pp. 2674–2679.
URL: *https://doi.org/10.1109/GLOBECOM54140.2023.10437193*

Chatterjee, A. and Prinz, A. (2022). Applying spring security framework with keycloak-based oauth2 to protect microservice architecture apis: A case study, *Sensors* **22**: 1703.
URL: *https://doi.org/10.3390/s22051703*

Chen, H., Pendleton, M., Njilla, L. and Xu, S. (2020). A survey on ethereum systems security: Vulnerabilities, attacks, and defenses, *ACM Comput. Surv.* **53**(3).
URL: *https://doi.org/10.1145/3391195*

Christidis, K. and Devetsikiotis, M. (2016). Blockchains and smart contracts for the internet of things, *IEEE Access* **4**: 2292–2303.
URL: *https://doi.org/10.1109/ACCESS.2016.2566339*

Kettani, H. and Carnley, P. (2019). Identity and access management for the internet of things, *International Journal of Computers, Communications Control (IJCCC)* **8**: 129–133.
URL: *https://doi.org/10.18178/ijfcc.2019.8.4.554*

Nuss, M., Puchta, A. and Kunz, M. (2018). Towards blockchain-based identity and access management for internet of things in enterprises, *in* S. Furnell, H. Mouratidis and G. Pernul (eds), *Trust, Privacy and Security in Digital Business*, Vol. 11033 of *Lecture Notes in Computer Science*, Springer, Cham.
URL: *https://doi.org/10.1007/978-3-319-98385-1*$_1$2

Priest, J., Cooper, C., Lovell, S., Shi, Y. and Lo, D. (2023). Design and implementation of an erc-20 smart contract on the ethereum blockchain, *2023 IEEE International Conference on Big Data (BigData)*, pp. 2334–2338.
URL: *https://doi.org/10.1109/BigData59044.2023.10386258*

Pöhn, D. and Hommel, W. (2023). New directions and challenges within identity and access management, *IEEE Communications Standards Magazine* **7**(2): 84–90.
URL: *https://doi.org/10.1109/MCOMSTD.0006.2200077*

Sadique, K., Rahmani, R. and Johannesson, P. (2023). Didm-eiotd: Distributed identity management for edge internet of things (iot) devices, *Sensors* **23**: 4046.
URL: *https://doi.org/10.3390/s23084046*

Sicari, S., Rizzardi, A., Grieco, L. and Coen-Porisini, A. (2015). Security, privacy and trust in internet of things: The road ahead, *Computer Networks* **76**: 146–164.
URL: *https://www.sciencedirect.com/science/article/pii/S1389128614003971*

Singla, A. and Bertino, E. (2018). Blockchain-based pki solutions for iot, *2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC)*, pp. 9–15.
URL: *https://doi.org/10.1109/CIC.2018.00-45*

Spadavecchia, G., Fiore, M., Mongiello, M. and De Venuto, D. (2024). A novel approach for fast and secure data transmission using blockchain and iot, *2024 13th Mediterranean Conference on Embedded Computing (MECO)*, pp. 1–4.
URL: *https://doi.org/10.1109/MECO62516.2024.10577932*

Williamson, G., Koot, A. and Lee, G. (2022). Non-human account management (v4), *IDPro Body of Knowledge* **1**(11).
URL: *https://doi.org/10.55621/idpro.52*

Wood, G. et al. (2014). Ethereum: A secure decentralised generalised transaction ledger, *Ethereum project yellow paper* **151**(2014): 1–32.

Zhao, Z. (2022). Comparison of hyperledger fabric and ethereum blockchain, pp. 584–587.
URL: *https://doi.org/10.1109/IPEC54454.2022.9777292*

Ziegeldorf, J., Morchon, O. and Wehrle, K. (2014). Privacy in the internet of things: Threats and challenges, *Security and Communication Networks* **7**.
URL: *https://doi.org/10.1002/sec.795*