Website:www.jriiejournal.com ISSN 2520-7504 (Online) Vol.9, Iss.3, 2025 (pp. 182 - 194)

NFT-Based Authentication Framework for Securing Enterprise Microservices in Big Data Environment

Saravanapavan Nasiketha[1] & Umayanga Athapaththu[2]
[1]NSBM Green University
[2]Plymouth University UK
Email: naji@nsbm.ac.lk/imanthaumayanga512@gmail.com

Abstract: *This study proposes a secure authentication framework based on Non-Fungible Tokens (NFTs) for enterprise microservices in big data environments. The objective was to overcome the limitations of traditional mechanisms such as JSON Web Tokens (JWTs), which lack built-in traceability, revocation, and protection against session-based attacks. Developed as a conceptual system design project, the framework was simulated using a custom private blockchain built in Go (Golang). NFTs were minted and transferred through smart contracts to act as temporary, verifiable access tokens. A custom hardware wallet, built on the ESP32-S3 microcontroller and programmed using the Arduino framework, was used to establish a physical barrier between the user and the system. This approach ensured that token ownership remained tamper-resistant and device-bound, ensuring that it remained secure. No human subjects were involved in this study; instead, the system was evaluated through functional test scenarios to assess authentication flow, session control, and token lifecycle management. Tools such as Docker and Redis were used to support simulation and deployment. Results indicated that the NFT-based system demonstrated improved resistance to token hijacking, session fixation, and replay attacks compared to JWT-based alternatives. Unlike conventional systems, it does not rely on static credentials and allows token revocation via on-chain burning and session-bound control. The study recommends future validation in real-world enterprise settings to assess scalability, fault tolerance, and real-time integration. The proposed framework offers a pathway toward decentralised, privacy-preserving authentication solutions suitable for secure and distributed environments.*

Keywords*: NFT, Authentication, Blockchain, Web3, Security, Microservices*

How to cite this work (APA):

Nasiketha, S. & Umayanga, A. (2025). NFT-Based Authentication Framework for Securing Enterprise Microservices in Big Data Environment. *Journal of Research Innovation and Implications in Education, 9*(3), 182 – 194. **https://doi.org/10.59765/jriie.9.3.18**,

1. Introduction

Modern authentication systems often depend heavily on static user credentials such as usernames and passwords. Even with enhancements like multi-factor authentication (MFA), attackers continue to exploit human error, phishing, and token reuse. Once compromised, JWTs remain active until expiration and can be reused without immediate detection. Though token blacklisting and reduced expiration times offer partial mitigation, these approaches are difficult to scale and manage effectively.

In June 2025, cybersecurity analysts reported one of the largest known aggregations of leaked credentials in history—over 16 billion username and password combinations exposed from multiple data breaches, including major platforms such as Apple, Google, Facebook, Netflix, and Microsoft (Winder, 2025). While not a breach of a single provider, this massive dataset of compromised credentials underscores the widespread vulnerabilities associated with centralised authentication models that rely on static user credentials. The global scale of this incident poses a severe risk to digital security across industries, especially in systems where credential reuse, password-only logins, and limited revocation controls persist. In developing regions such as Sri Lanka**,** the consequences of such leaks can be even more damaging due to the limited adoption of zero-trust models and advanced identity protection mechanisms. Enterprises, educational institutions, and government agencies in these contexts often lack the infrastructure for decentralised or hardware-enforced authentication. This study responds to this urgent global and local challenge by proposing an NFT-based authentication system that introduces a physically enforced, revocable, and session-bound mechanism for secure access, designed to resist credential leakage, session hijacking, and replay attacks even in distributed or microservice-based environments.

Studies such as Hannousse and Yahiouche (2020) and Barabanov and Makrushin (n.d.) have highlighted these vulnerabilities in JWT-based architecture, particularly in microservice environments. Blockchain technology has emerged as a potential solution, offering decentralised, tamper-proof, and verifiable identity mechanisms. However, even blockchain-based models—such as cryptographic key-pair systems or static hash methods—have limitations, including performance overhead, weak revocation, and limited integration with physical device security.

This study introduces an innovative authentication framework that leverages Non-Fungible Tokens (NFTs), private blockchain networks, and hardware wallets to address these challenges. By assigning session-bound NFTs to physical devices during login, the system creates a verifiable and tamper-resistant authentication artefact. Unlike conventional systems, this method establishes a physical barrier between the user and system access, minimising token theft and session hijacking risks. Authentication sessions are secured through token minting and revocation managed by smart contracts, offering traceability and on-chain auditability.

This framework is especially relevant for enterprise environments operating within big data infrastructures or Zero Trust Architectures, where sensitive services require decentralised and fail-proof access control. Although developed as a conceptual model, the system was evaluated through simulated test cases using a private blockchain and a hardware wallet using ESP32–S3.

The purpose of this study is to design and conceptually evaluate an NFT-based authentication mechanism that overcomes the limitations of JWT-based systems by enhancing traceability, token lifecycle management, and user-device integrity.

Cybersecurity has become a top priority for organisations worldwide in today's increasingly interconnected digital landscape. As technology evolves, so do the threats targeting it. Organisations face a constant barrage of cyberattacks, many of which exploit vulnerabilities in authentication systems—the gateways to sensitive information. Therefore, ensuring the security of user identities and access credentials is a cornerstone of digital security strategy. Among the myriad cybersecurity concerns, authentication vulnerabilities represent one of the most pressing challenges. Cybercriminals continue to exploit weaknesses in traditional systems to gain unauthorised access, leading to data breaches, financial loss, and reputational damage. This has necessitated the development of more secure and resilient authentication methods that can withstand modern attack vectors.

Most modern web applications utilise JSON Web Tokens (JWTs) for authentication, especially in environments based on JSON-based data exchanges. JWTs are popular due to their stateless nature, which eliminates the need for server-side session storage and facilitates efficient performance across distributed systems. However, despite these operational benefits, JWTs are not without critical security flaws. One of the most significant concerns with JWTs is their vulnerability to social engineering attacks. In many cases, attackers exploit human behaviour rather than technical flaws. Research consistently shows that privileged users, such as system administrators or executives, are prime targets. Through deception and manipulation, attackers can obtain user credentials or access valid JWTs, enabling unauthorised access to systems without triggering alarms. If multi-factor authentication (MFA) is not enforced, these attacks become even more dangerous, as the compromised tokens can be used without further verification.

Beyond their susceptibility to social engineering, JWTs also suffer from a lack of intrinsic traceability. Once issued, a token remains valid until its expiration, regardless of whether it has been stolen or misused. This stateless characteristic makes it difficult to detect unauthorised use in real time. While mitigation strategies—such as maintaining token blacklists or reducing token lifespan—exist; they add complexity to large-scale systems and often fail to deliver comprehensive security.

Furthermore, JWTs are typically stored locally on the client side using browser storage mechanisms, cookies, or session storage—all of which can be compromised. This local dependency, combined with poor revocation and auditing capabilities, creates a high-risk environment for enterprise applications and services.

To address these persistent security concerns, a transformative approach to authentication is necessary. This research proposes a novel authentication framework based on Non-Fungible Tokens (NFTs), leveraging blockchain technology to create a secure, decentralised, and traceable authentication system. NFTs, by their very nature, are unique, immutable, and verifiable. When used in authentication, these attributes provide substantial security benefits over traditional tokens. Unlike JWTs, NFTs can be issued, tracked, transferred, and revoked through blockchain transactions, offering a transparent audit trail that cannot be altered retroactively. This eliminates the risks associated with static tokens and local storage vulnerabilities.

The adoption of an NFT-based authentication system introduces a paradigm shift in how digital identities are verified. By eliminating the need for user-managed passwords and central session storage, this model minimises the attack surface and enhances resistance to social engineering. The blockchain's inherent transparency and immutability ensure that all authentication events are permanently recorded, enabling real-time monitoring and incident response. This approach aligns with the principles of Web3, which emphasise decentralisation, privacy, and user ownership of data. By integrating NFT-based authentication into enterprise systems— especially those built on microservice architectures and operating within big data ecosystems— organisations can significantly elevate their security posture while embracing the future of decentralised identity management.

2. Literature Review

OAuth 2.0 and OpenID Connect are two of the most popular approaches; they offer strong support for user authentication and delegated permission, respectively. These protocols are often used with JSON Web Tokens (JWT) to securely provide identification information in a concise, stateless fashion across services. Furthermore, by functioning as middlemen that handle access control and token validation, API Gateways frequently operate as enforcement points for these processes. Mutual Transport Layer Security, or MTLS, is used in service-to-service communication to encrypt and authenticate communications between microservices. Even though these approaches are widely used, integrating them across dispersed components adds complexity and expands the attack surface of the system, highlighting the significance of centralised identity management systems and established security practices. (de Almeida & Canedo, 2022)

JWT's stateless nature and flexibility in identification and access management, token-based authentication, in particular, JSON Web Tokens (JWT), are widely used. Although traditional JWT implementations are often used in online applications, they have drawbacks such as exposing sensitive data and susceptibility to abuse and eavesdropping.(Barabanov & Makrushin, n.d.; Diaz Rivera et al., n.d.; Hannousse & Yahiouche, 2020)

By using a static hash-based approach, blockchain technology provides a decentralised and secure authentication method. This architecture guards against man-in-the-middle attacks by keeping a persistent record of user credentials. Token revocation and dynamic access control are absent from this method, though. For tamper-proof access history, the hybrid authentication paradigm mixes public and private blockchain networks; however, significant transaction fees cast doubt on the concept's scalability and viability for large-scale implementations. (Diaz Rivera et al., 2023)

2.1 Vulnerabilities of the JWT-based authentication system

Despite the critical role of authentication systems in securing applications, they remain vulnerable to a range of sophisticated threats, especially in microservice-based architectures. One prominent challenge is the increased attack surface due to service decomposition, where each microservice must authenticate independently, leading to inconsistent enforcement and potential configuration errors. Edge-level authentication, if not complemented by service-level checks, becomes a single point of failure and may violate the "defence in depth" principle. Token-based approaches, such as JWTs, are susceptible to interception or leakage if not properly secured and validated, potentially enabling privilege escalation. Additionally, improper implementation of service-to-service authentication or the misuse of self-signed identity propagation methods can result in impersonation attacks, especially when mutual trust between services is assumed without rigorous verification. These vulnerabilities underscore the necessity of layered security strategies, continuous policy verification, and robust key management to maintain the integrity of authentication systems. (Barabanov & Makrushin, n.d.)

Microservice architectures (MSAs) introduce unique vulnerabilities to authentication systems due to their distributed and loosely coupled nature. Key vulnerabilities include unauthorised access, where attackers exploit weak or improperly implemented authentication mechanisms, and session management flaws, which can allow session hijacking or token theft. The large number of entry points in MSAs increases the attack surface, making it challenging to enforce consistent authentication across all services. Additionally, the reliance on third-party identity providers or untrusted microservices can introduce risks of misconfiguration or compromised credentials. Inadequate encryption of authentication tokens during inter-service communication further exacerbates the potential for interception and misuse, necessitating robust access control and continuous monitoring to mitigate these threats. (Hannousse & Yahiouche, 2020)

In microservice architectures (MSAs) implementing Zero Trust Networking (ZTN), authentication systems face significant vulnerabilities, particularly in the secure delivery of enrolment tokens such as JSON Web Tokens (JWTs). The distributed nature of MSAs creates multiple entry points, increasing the risk of unauthorised access due to weak or inconsistent authentication mechanisms across services. A critical vulnerability lies in the insecure transmission of JWTs, as their base64url encoding lacks robust encryption, making them susceptible to eavesdropping or tampering if not properly secured. Additionally, the absence of strong ownership verification for one-time tokens (OTTs) allows any entity possessing a token to initiate enrolment, potentially generating valid X.509 certificates for unauthorised access to the Zero Trust network. Reliance on third-party identity providers or untrusted microservices further introduces risks of misconfiguration or credential compromise. The proposed blockchain-based solution mitigates some risks by leveraging non-fungible tokens (NFTs) for proof of ownership and encryption of JWT metadata, but vulnerabilities persist due to potential smart contract exploits, high transaction costs, and the need for precise synchronisation in private blockchain operations, which could disrupt validation processes if mismanaged. (Diaz Rivera et al., n.d.)

2.2 Existing blockchain-based solution for authentication

One approach replaces traditional passwords with cryptographic key pairs to improve security and decentralisation. This system uses smart contracts for role-based access control (RBAC), dynamically assigning and managing roles without reliance on centralised servers. It integrates multi-factor authentication (MFA) methods, such as biometrics and one-time passwords (OTPs), to strengthen user verification. Decentralised identity management allows users to securely control their credentials without third-party involvement, reducing the risk of centralised data breaches. However, implementation complexity and performance overhead in high-traffic environments remain challenges (Ukani et al., 2023)

A decentralised semantic identity management solution combines WebID, JSON Web Tokens (JWTs), and blockchain technology. WebID profiles are stored securely on the InterPlanetary File System (IPFS) and linked to immutable records on the Namecoin blockchain. JWTs facilitate stateless authentication, enhancing privacy and security by reducing reliance on centralised servers. While this approach improves user privacy and data integrity, it does not address dynamic token revocation or real-time access control, limiting its applicability in dynamic MSAs (Faísca & Rogado, 2016)

A static hash-based authentication model stores a permanent record of user credentials on the blockchain, providing robust defence against man-in-the-middle attacks through cryptographic key verification. This approach ensures tamper-proof credential storage but assumes permanent validity of registered hashes, lacking mechanisms for dynamic access control or token revocation. As a result, it is less suitable for environments requiring frequent updates to access permissions (Diaz Rivera et al., 2023)

A hybrid model combines public and private blockchain networks for authentication. User identities are stored on a public blockchain, such as Ethereum, with validation performed through private Hyperledger Fabric networks. This system maintains a tamper-proof access history, enhancing auditability. However, high transaction fees on public blockchains, particularly Ethereum, raise scalability concerns for large-scale deployments. Additionally, the model lacks detailed mechanisms for efficient token distribution and revocation (Diaz Rivera et al., 2023)

2.3 NFT-Based Authentication for Zero Trust Networking

A dynamic blockchain-based solution uses non-fungible tokens (NFTs) instead of traditional JWTs for authentication within a Zero Trust framework. The authentication token, encoded as an NFT, is encrypted with the user's blockchain public key and mapped to their public blockchain address, ensuring secure ownership verification. Smart contracts enable token revocation by "burning" tokens after use or expiration, preventing reuse and enhancing security compared to JWT systems. Despite these advantages, the system introduces significant delays in token issuance and faces performance challenges in real-time authentication scenarios. Integration with ZTN ensures strong identity verification, but high computational overhead and blockchain transaction costs remain barriers to scalability (Diaz Rivera et al., 2023)

2.4 Auditing and Monitoring Limitations

Despite advances in auditing and monitoring, many blockchain-based authentication systems lack robust mechanisms for mitigating security breaches post-compromise. Auditing typically tracks user and service actions, but compromised JWTs offer limited recourse beyond logging misuse. Machine learning-based anomaly detection and proactive monitoring solutions have been proposed to identify unauthorised access, but their adoption is hindered by implementation complexity and insufficient real-world testing. These limitations highlight the need for improved response capabilities in blockchain-based systems (Barabanov & Makrushin, n.d.; Diaz Rivera et al., 2023; Hannousse & Yahiouche, 2020)

2.5 Usage of NFTs in authentication systems

NFTs are used for authentication in a blockchain-based system, guaranteeing safe encryption linked to the user's identity. It provides security against token interception and misuse when integrated into a zero-trust networking infrastructure; however, it causes delays in token issuance. Additionally, it uses smart contracts to implement a token revocation mechanism; nevertheless, in real-time authentication settings, it encounters performance issues. (Diaz Rivera et al., 2023)

Many systems lack adequate means for addressing security vulnerabilities after they occur, despite advancements in auditing and monitoring approaches. While prevention and ongoing auditing are the main topics of the current study, reaction skills are frequently lacking. Usually, auditing keeps track of user and service activity, but if a JWT is hacked, there isn't much that can be done except record the abuse. (Diaz Rivera et al., 2023)

The absence of real-world testing and the difficulty of implementation have hindered the adoption of machine learning-based anomaly detection and other proactive monitoring systems. According to recent research, blockchain technology can enhance authentication in microservice architectures by substituting Non-Fungible Tokens (NFTs) for JWTs, offering traceable transaction history and cryptographically verifiable ownership, and resolving problems like replay attacks and token theft. (Barabanov & Makrushin, n.d.; Hannousse & Yahiouche, 2020)

The scalable architecture known as Blockchain-based Multi-Factor Authentication-as-Service (BMFAaaS) is ideal for dispersed systems like fog computing and the Internet of Things. It emphasises improved scalability via defined protocols, smooth interaction with current systems, and universal accessibility. In addition to offering domain-specific

Applications like decentralised identity management for education and IoT device authentication, the solution employs blockchain consensus processes to provide safe authentication while protecting privacy and reducing typical security risks. (Almadani et al., 2023)

Recent research shows how blockchain technology might improve microservice architecture authentication systems, especially when Non-Fungible Tokens (NFTs) are used in place of JWTs. Key problems in JWT-based systems, such as token theft, replay attacks, and revocation difficulties, are addressed by NFTs, which provide cryptographically verified ownership and a traceable transaction history. NFTs guarantee that every access attempt or token transfer can be safely tracked by logging authentication events on a blockchain, creating an unchangeable audit trail. (Barabanov & Makrushin, n.d.; Hannousse & Yahiouche, 2020)

Additionally, blockchain-based solutions make decentralised access control possible, which lessens the need for a central authority to validate tokens. Because each service can independently validate token validity without central coordination, this provides superior scalability in contexts with significant microservice traffic. Nevertheless, there are still issues with the performance optimisation of these blockchain-based systems, especially when it comes to reducing overhead in high-throughput settings. (Barabanov & Makrushin, n.d.; Hannousse & Yahiouche, 2020)

2.6 Research Gap and Justification

Despite these advancements, existing authentication systems do not integrate NFTs with physical hardware enforcement**.** Most models rely solely on digital verification without ensuring physical user presence. Additionally, no reviewed studies have implemented a dual private blockchain setup to manage the token lifecycle and enforce real-time revocation with hardware wallet integration. This creates a critical gap in both security and auditability, particularly for high-stakes environments such as enterprise microservices or big data platforms.

Furthermore, current literature focuses heavily on Western technological ecosystems. There is limited exploration of these systems in developing or emerging regions, where risks like credential leaks, weak infrastructure, and social engineering are even more pronounced. Addressing these issues through a 02 is the focus of this study.

3. Methodology

This study adopted a conceptual system design and simulation-based research approach to develop and evaluate an NFT-based authentication framework. The system was designed, implemented, and tested in a controlled virtual environment using blockchain simulators and local network components to assess security features and token lifecycle management.

3.1 Development methodology

This section details the choices for back-end, blockchain integration, and hardware wallets, and how these technologies work together to achieve the desired outcomes.

Technology adopted. A thorough study of available resources and technologies was conducted to determine the most suitable solutions for developing a robust and secure user authentication service.

The designed system consists of multiple components that work together to provide secure and efficient user authentication. The key components include the database, blockchain integration, back-end services, and hardware wallets.

Database: The database is responsible for storing user information, authentication logs, and other critical data. The chosen database system is Hyperledger Indy in Blockchain. This provides a robust and flexible storage solution with the capabilities of blockchain.

Blockchain Integration Blockchain technology is used to manage the dynamic ownership of NFTs that represent user authentication credentials.

- Smart contracts on the blockchain handle the ownership transitions between the company and employees.
- Smart Contracts: Developed in Solidity, these contracts manage NFT ownership and authentication logs.
- NFT Ownership: Represents user credentials, dynamically changing ownership between the company and employees during login and logout processes.
- MFA – Multi Factor Authentication -Time-based one time password (OTP): generate a time-based OTP system to verify whether the user is an authorised user.
- Back-End Services The back-end services handle business logic, data processing, and communication with the blockchain.
- Blockchain server: after the authentication process client will receive the request token from the back-end server, and it will be stored in the wallet Hardware Wallets

3.2 Technologies used

The back-end is responsible for the business logic, data storage, and integration with the blockchain. The chosen technologies should support multiple users, efficient data handling, inherent security features, easy data retrieval, and maintenance.

Blockchain technology is integral to managing dynamic NFT (Non-Fungible Token) ownership for authentication. Smart contracts on the blockchain are developed to securely and transparently handle ownership transitions.

By maintaining privet blockchain, companies can reduce the transaction cost and computation cost-effectively. For development and testing purposes, you can use the Ganache application, and for local blockchain can use the Ethereum blockchain.

- Smart Contracts Ownership Management Contract: Handles the dynamic ownership of NFTs between the company and employees. Use Solidity to develop the smart contracts using solidity.
- Golang: Known for its performance, lightweight and efficiency in handling concurrent processes, Golang is chosen for building microservices that handle business logic and API interactions. Using suitable frameworks, organisations can improve their performance and security further.
- Authentication Log Contract: Records login and logout events, ensuring immutability and transparency. For the development of smart contracts, the Solidity language is specially designed for the development of smart contracts. Hardware Wallet and in-memory database with Multi-Factor Authentication. Hardware wallets are integrated to provide an additional security layer, requiring physical confirmation for critical operations.
- Multi-Factor Authentication (MFA): To improve security, add the OTP.
- Nginx: Used as a reverse proxy to manage SSL termination and load balancing. Implementing the Reverse proxy and disabling the forward proxy can improve security further and more efficiently.
- Hardware wallet: to store the NFTs, it needs to use a hardware wallet.

3.3 System Architecture

Existing JWT-based authentication systems are vulnerable to token hijacking and centralised security breaches in enterprise microservices and big data environments. This project proposes a blockchain-integrated NFT-based framework as a secure, decentralised alternative to overcome these limitations.

The proposed authentication system leverages blockchain technology, NFTs (Non-Fungible Tokens), and conventional MFA (Multi-Factor Authentication) to replace traditional JWT-based models, addressing security, privacy, and token lifecycle management challenges. This architecture introduces a decentralised, tamper-proof framework that enhances both user and service authentication in microservice-based applications.

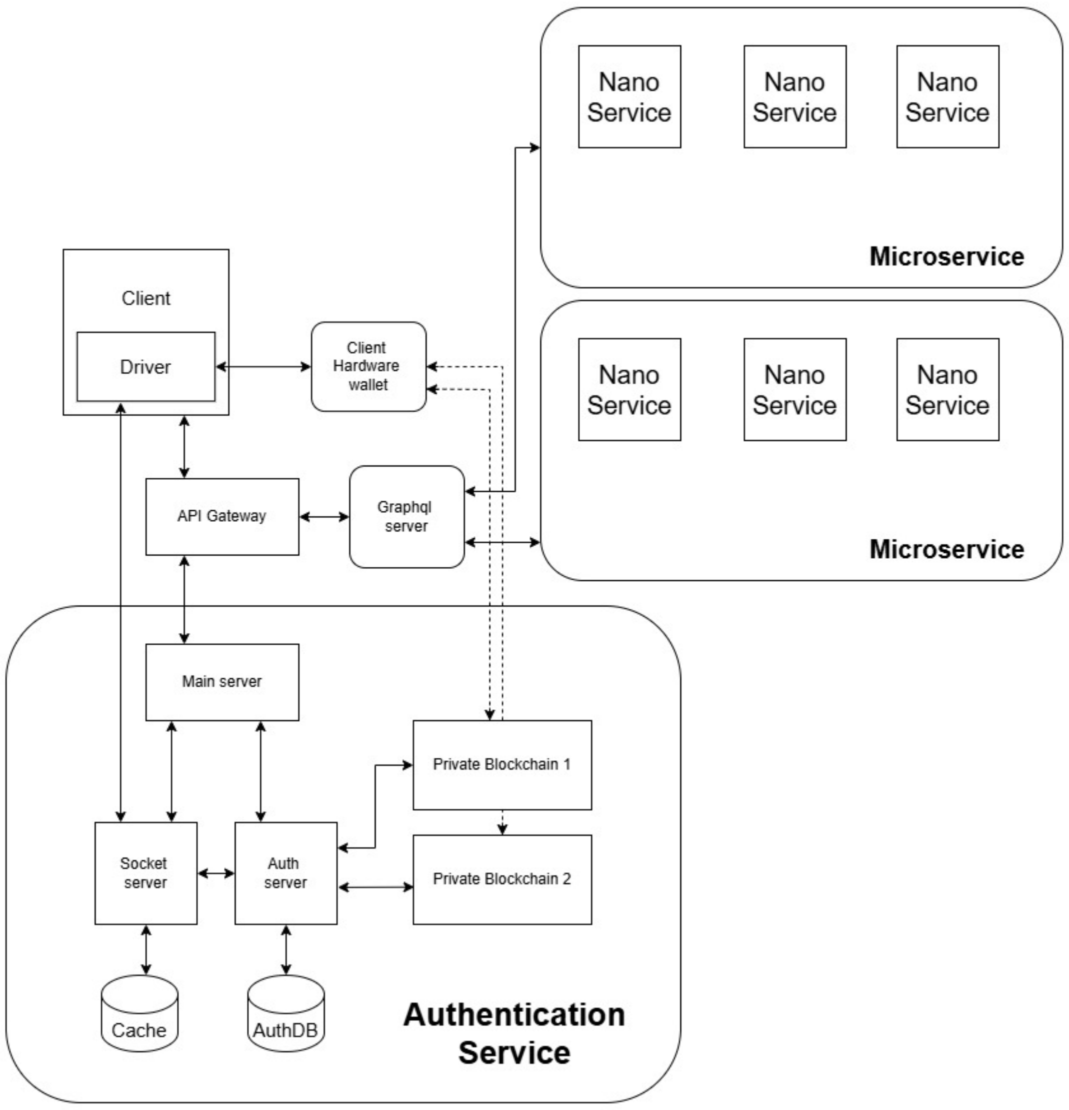

Figure 1: System architecture overview

The system architecture is composed of multiple modular layers, each responsible for handling a specific part of the authentication and service communication process. These include the Client Interface, API Gateway, GraphQL Server, Authentication Service, Private Blockchains, and the Microservices Layer.

- Dual Blockchain Setup with API Integration: At the heart of the system is a dual blockchain architecture designed to separate private interactions from sensitive, internal operations. This setup ensures security, transparency, and fault tolerance. The blockchain layer is exposed through a set of well-structured APIs, allowing seamless integration with external services and internal modules. This delivers a robust foundation for NFT creation, validation, and secure token-based transactions, enabling efficient and tamper-proof authentication workflows.
- Main Backend Server API for User Request Handling: The backend server serves as the central hub for processing user requests. It manages key operations such as login requests, NFT validation, token issuance, and user session control. This API layer ensures efficient communication between the client-side applications and blockchain infrastructure while upholding the security policies defined within the system.
- Socket Server for Server–Hardware Wallet Communication: A persistent and efficient communication mechanism is established between the server and the custom hardware wallet through a socket server. This server maintains a secure channel for real-time interactions such as NFT transfers, wallet state updates, and session validations. It plays a vital role in ensuring low-latency, secure operations

between the server infrastructure and the client's physical wallet.

- Custom-Designed Hardware Wallet for NFT Management: The project delivers a proprietary hardware wallet developed specifically for managing user authentication NFTs. This wallet temporarily holds NFT-based credentials and interacts securely with the server during authentication sessions. Its design ensures that NFTs cannot be extracted, cloned, or misused, significantly enhancing the physical security of the authentication process. To develop the hardware wallet, use the ESP32-S3 model.
- Client-Side Software for Secure Client–Server Communication: To complete the end-to-end authentication ecosystem, a lightweight client application is developed to handle communication between the user and the server infrastructure. This software facilitates login operations, NFT synchronisation, and local hardware wallet interaction. It ensures a seamless and intuitive user experience while enforcing all required security protocols in real-time.

This authentication system creates a physical barrier between the user and the system while maintaining the same user experience. Because of the NFT usage proposed system does not depend on the user credentials.

3.4 Proposed System Workflow

This section outlines the detailed process of the proposed NFT-based authentication system, highlighting how NFTs, private blockchains, MFA, and secure communication protocols are orchestrated to deliver robust, decentralised authentication.

3.4.1 User Login and Wallet Connection

The authentication process begins when the user attempts to log into the system. A prerequisite for access is the connection of the user's wallet (preferably a hardware wallet). The system verifies the presence of this wallet during login. If no wallet is connected, the login request is denied, ensuring that all authentication attempts originate from authorised hardware.

3.4.2 Credential Submission and Multi-Factor Authentication (MFA)

Once the wallet is detected, the user submits their login credentials. The server then initiates a multi-factor authentication (MFA) protocol, leveraging time-based OTPs or other second-factor methods. This ensures that even if credentials are compromised, unauthorised users are blocked from proceeding without possessing the physical authentication device.

3.4.3 NFT Revocation and Ownership Transfer

Upon successful credential and MFA verification, the system transfers ownership of the NFT from the user's wallet to the company. This NFT, acting as a session token, is revoked and securely recorded in Blockchain 1 (Request Blockchain). This marks the termination of the user's current authentication session and prevents reuse of the same token for future requests.

3.5 Token Termination Logging

Blockchain 1 records the token burn or revocation event immutably, ensuring that any subsequent attempt to reuse the invalidated token will be cryptographically rejected by the system.

3.6 Issuance of a New Authentication Token

A new, valid NFT is minted and transferred to the user's wallet. This NFT functions as a secure, blockchain-bound authentication token, replacing traditional mechanisms like browser cookies or local session storage. The minting and assignment of this token are recorded in Blockchain 2 (Auth Blockchain), introducing a secondary layer of validation and tamper resistance.

3.7 Request-Level Validation Using NFTs

Subsequent user requests are authenticated by verifying the presence and ownership of a valid NFT. Unlike traditional JWT-based systems, where tokens are stored locally and prone to theft, this approach ensures that only the authenticated user in possession of the corresponding wallet can interact with protected services.

3.8 Proxy Architecture and Load Balancing

The system employs a reverse proxy and a load balancer to manage inbound traffic efficiently, distribute load evenly across services, and optimise system performance. In parallel, a forward proxy manages outbound traffic, enabling secure and efficient communication with external or third-party services when required.

3.9 VPN and End-to-End Encryption

To protect communication channels between client and server, a Virtual Private Network (VPN) tunnel is established, encrypting all transmitted data. Additionally, SSL/TLS encryption is implemented between microservices and system layers to ensure data confidentiality and integrity during transit.

3.10 Logout and Secure Session Termination

When the user logs out, the current NFT is returned to the server, where ownership is transferred back to the company, and the token is subsequently burned. This event is logged in the blockchain, ensuring that the session termination is secure, verifiable, and immutable.

3.11 Holistic Security through Integrated Technologies

By combining NFTs, private blockchains, MFA, hardware wallet authentication, and layered cryptographic protections, the system offers a decentralised, tamper-proof authentication model. This approach addresses the core vulnerabilities of conventional JWT-based authentication—including token theft, replay attacks, and local storage manipulation—while enhancing privacy and system integrity.

3.12 Smart Contract Logics

The smart contract acts as the foundational backend logic for the NFT-based authentication system. It governs the entire lifecycle of authentication tokens, including minting, validation, and revocation. Written in Solidity and deployed on a local or private Ethereum-compatible blockchain (e.g., Ganache), the contract adheres to the ERC-721 standard to ensure compatibility and uniqueness of issued tokens.

The smart contract handles the following core responsibilities:

- Minting Authentication NFTs: Upon successful user authentication, the backend system (controlled by the company) invokes the smart contract to mint a new ERC-721 token. This token serves as a unique and verifiable session credential. The NFT is minted directly to a temporary, ephemeral wallet associated with the current session.
- Ownership and Transfer Control: To ensure the security and integrity of the authentication process, the smart contract restricts the transfer of NFTs. Once assigned, the token cannot be transferred by the user to another wallet. This prevents credential sharing and enforces session isolation.
- Token validation: The smart contract provides functions that allow external services (e.g., microservices in a distributed system) to verify whether a given wallet address holds a valid, unrevoked authentication NFT. This supports secure, decentralised authentication checks.
- Token Revocation (Burning): When a user logs out or their session is terminated, the corresponding NFT must be destroyed. The smart contract includes a function to burn the token, permanently invalidating it. Only authorised entities (e.g., the backend or system owner) can initiate this burn operation.

To ensure robustness and minimise vulnerabilities, the smart contract implements the following security best practices:

- Role-Based Access Control: Only the system owner (i.e., the company backend) is authorised to mint or burn tokens. This is enforced using onlyOwner or role-based permission structures (e.g., OpenZeppelin's AccessControl).
- Reentrancy Protection: The contract avoids calling external contracts or untrusted addresses during critical operations, mitigating the risk of reentrancy attacks.
- Overflow and Underflow Protection: Using Solidity version $\geq$ 0.8 automatically includes overflow checks, ensuring safe arithmetic operations during token counter increments and validations.
- Input Validation: All critical functions include checks to validate token ownership and state before performing operations, minimising the risk of unauthorised access or manipulation.

The smart contract encapsulates the core logic for secure, revocable, and session-bound authentication using NFTs. It ensures that authentication credentials are:

- Unique and tamper-proof
- Assigned only by authorised systems
- Bound to a session-specific wallet
- Automatically invalidated upon logout

By handling authentication token lifecycle on-chain, the system introduces a decentralised and transparent mechanism for secure access management, particularly well-suited for modern microservice architectures and distributed systems.

3.13 Implementation

This section outlines the complete setup process for both development and production environments of the authentication system. It includes the installation of necessary tools, configuration of the blockchain network, hardware wallet preparation, and backend and client software deployment. Each stage has been prioritised to ensure a smooth setup process for developers and system integrators.

3.13.1 Minimum Platform Requirements

To install and operate the system effectively, the following minimum hardware and software specifications are required:

- Operating System: Linux (Ubuntu 20.04+), macOS 12+, or Windows 10+
- RAM: Minimum 8 GB (16 GB recommended)
- Processor: 64-bit, multi-core
- Storage: Minimum 10 GB free disk space

Software Dependencies:

- Docker (v20.10 or above)
- Python (v3.9 or above)
- Go (v1.18 or above)
- Redis (for socket server)
- PostgreSQL or AWS QLDB (for backend storage)

3.13.2 Development Environment

For local testing, the system supports the use of Ganache, a personal blockchain that simulates Ethereum-like behaviour. This allows for rapid testing and debugging without connecting to a live network.

For advanced local simulations or custom blockchain behaviour, a mock blockchain API or a lightweight, Go-based simulator can be configured.

3.13.3 Production Environment

In production, it is recommended to use AWS Managed Blockchain (AMB) with Hyperledger Fabric. AMB allows the creation of a private, permissioned blockchain network suitable for enterprise-grade security and scalability. Each node and smart contract should be carefully configured and deployed according to AWS's best practices.

Smart contracts can be developed using Solidity or Go, depending on the chosen blockchain platform. After development, they must be tested rigorously before deployment.

3.13.4 Smart Contract Deployment

Before integrating the blockchain with the rest of the system, smart contracts must be:

- Developed in Solidity/Go,
- Deployed to the test network
- Then deployed to the production blockchain network (AMB Hyperledger).

The contracts must manage authentication tokens, wallet ownership, and token burning functionalities as defined in the system architecture.

3.13.4 Hardware Wallet Integration

1. **Development Phase**

In development, the system uses an ESP32-S3 N16R8 microcontroller programmed via the Arduino framework. Before deployment:

- Assign a secure password to each microcontroller.
- Configure the microcontroller to simulate wallet operations, including receiving tokens, interacting with the API, and securely erasing session data.

2. **Production Phase**

For production, use a dedicated hardware wallet that supports multi-account storage and secure private key management. These wallets should be capable of secure communication with the backend and locally connected machines.

The wallet firmware must include all necessary business logic, and communication with the backend must follow secure protocols (e.g., encrypted communication over USB or Bluetooth).

3.14 Wallet Initialisation and User Assignment

Each user must be assigned a unique wallet. The steps are as follows:

1. Create a new wallet instance on the blockchain network.
2. Assign this wallet to the user's hardware device.
3. Upload the required configuration and authentication logic to the hardware wallet or microcontroller.

This step ensures a one-to-one mapping between blockchain accounts and physical wallets, enhancing security and traceability.

3. Backend and Database Setup

The main backend server handles authentication, blockchain interactions, and token lifecycle management. Setup includes:

- Application Backend Server: Built in Go/Python.
- Database: Use AWS QLDB (recommended for immutable logs) or PostgreSQL for relational data.
- Socket Server: For real-time communication, set up a Node.js or Python-based socket server with Redis as the in-memory database.

These components must be properly connected and secured using environment variables and secure configuration files.

4. Client Software Installation

Each user's machine must have the client software installed. This client application handles:

- Local hardware wallet detection,
- User interface interactions,
- Real-time communications with the backend,
- Password prompt and authentication workflow.

It should be designed for cross-platform use and support system tray notifications and modal popups for a seamless user experience.

3.15 Hardware Wallet Detection and Authentication

Once all components are in place:

1. The user plugs the hardware wallet into the machine.
2. The client software automatically detects the device.
3. A system notification is triggered.
4. A secure password prompt appears for the user to unlock the wallet.

Upon successful authentication, the session is initialised, and the system allows access to protected resources or services.

The setup process ensures a secure, scalable, and modular system that leverages blockchain technology for authentication. Both development and production environments are carefully structured to support iterative testing and robust deployment. Following this guide will enable any developer or system administrator to set up and operate the platform confidently.

4. Results and Discussion

The evaluation of the proposed NFT-based authentication system is based on conceptual analysis and controlled test scenarios, focusing on security, token lifecycle management, and system behaviour. While no real-world or performance benchmarking data was used, the functional behaviour of the system was assessed through simulated use cases in a development environment using local blockchain infrastructure (Ganache) and ephemeral in-memory wallets.

4.1 Evaluation Criteria

The system was evaluated based on the following aspects:

- Authentication Process Flow
- Token Revocation Mechanism
- Security Against Common Attacks
- Scalability in Concept
- Comparison with Traditional JWT-based Authentication

A. Simulated Test Cases

The table below outlines simulated functional test cases executed in a controlled environment.

Table 1: Simulated functional test cases

Test Case	Description	Expected Outcome
Successful Login	The user is assigned an ephemeral wallet and issued an NFT	Access granted with valid session
Replay Attack Attempt	Reuse of the same NFT after logging out	Access denied (NFT already burned)
Token Revocation (Logout)	Logging out triggers token burn	NFT removed, session invalidated
Unauthorised Access Attempt	Tampered or invalid NFT used in the request	Access blocked
Multiple Sessions Management	Multiple users accessing concurrently with unique wallets	Sessions managed independently

4.2Comparative Analysis: JWT vs NFT-Based Authentication

The following table provides a conceptual comparison between JWT-based authentication and the proposed NFT-based approach:

Table 2: JWT-based vs NFT-based authentication

Criteria	JWT Authentication	NFT-Based Authentication
Token Theft Resistance	Moderate (tokens can be intercepted)	High (NFT is session-bound, wallet-specific)
Replay Attack Protection	Weak (requires custom nonce logic)	Strong (burned NFTs are invalid)
Token Revocation	Requires blacklist or token invalidation tracking	Simple and reliable (burn NFT)

Authentication Flow	Stateless, fast	Stateful in design but secure
Integration Complexity	Low	Moderate (requires smart contract integration)
Scalability	High	High in private blockchain environments

4.3 Security Considerations

The NFT-based system improves security in the following ways:

- Session Isolation: Each login generates a new ephemeral wallet, reducing the impact of key compromise.
- NFT Ownership Verification: Ensures only the rightful holder of the wallet can present a valid NFT.
- Immediate Revocation: NFT can be burned at logout or timeout, immediately invalidating the session.
- No Long-Term Secrets Stored: The Ephemeral nature of wallets avoids long-term token storage or refresh issues.

This evaluation demonstrates that the NFT-based approach offers significant security and session control improvements over traditional token-based systems like JWT. The system is especially suited for use cases requiring short-lived, revocable, and non-transferable authentication mechanisms, such as in microservice architectures or distributed environments.

4.4 Discussion

NFT-based authentication system enhances security in big data environments by replacing traditional methods with decentralised identity verification using blockchain technology, ensuring tamper-proof and secure transactions.

4.4.1 Challenges in Developing an NFT-Based Authentication System

Blockchain-specific limitations require selecting the right blockchain, such as Hyperledger Fabric, to ensure efficient NFT support. NFT lifecycle management must include secure transfers and controlled burning using Role-Based Access Control (RBAC). Smart contract deployment and upgrades can be streamlined using proxy contracts, allowing seamless updates. Performance and latency issues can be addressed through Layer 2 solutions like Polygon zkEVM or off-chain verification. Security risks and compliance concerns necessitate thorough security audits and compliance-friendly logging mechanisms. Lastly, integration with existing identity systems can be achieved by providing API gateways and adopting decentralised identity standards.

4.4.2 Future Developments

Advanced biometric verification, including fingerprint and iris scanning, enhances security by providing robust identity authentication. Real-time data processing utilises AI-driven analytics for proactive threat detection. Live video authentication offers a dynamic method for verifying user identities in high-security environments. Government integration enables seamless verification by linking with national ID registries. AI-driven threat detection leverages machine learning for continuous security monitoring, identifying unusual patterns and potential risks. Adaptive security measures use AI to implement risk-based authentication, adjusting security levels based on user behaviour. Lastly, decentralised identity management empowers users with greater control over their data privacy, reducing reliance on centralised authentication systems.

5. Conclusion and Recommendations

The NFT-based authentication system presents a highly secure and decentralised alternative to traditional authentication mechanisms, such as JWT tokens. By leveraging managed blockchain services, the system ensures scalability, security, and ease of maintenance while reducing infrastructure overhead. Despite its advantages, the development process comes with challenges, including blockchain latency, smart contract management, ephemeral wallet handling, and compliance considerations. To address these, the careful selection of blockchain platforms, the implementation of secure smart contracts, and the optimisation of off-chain verification are crucial. Overall, this authentication model enhances identity security, mitigates risks like credential leaks and social engineering attacks, and improves trust in authentication for microservice applications in big data environments. With the right optimizations, it has the potential to revolutionize authentication standards, making identity verification more secure, tamper-proof, and efficient.

References

Almadani, M. S., Alotaibi, S., Alsobhi, H., Hussain, O. K., & Hussain, F. K. (2023). Blockchain-based multi-factor authentication: A systematic literature review. In *Internet of Things (Netherlands)* (Vol. 23). Elsevier B.V. https://doi.org/10.1016/j.iot.2023.100844

Barabanov, A., & Makrushin, D. (n.d.). *AUTHENTICATION AND AUTHORIZATION IN MICROSERVICE-BASED SYSTEMS: SURVEY OF ARCHITECTURE PATTERNS*.

de Almeida, M. G., & Canedo, E. D. (2022). Authentication and Authorization in Microservices Architecture: A Systematic Literature Review. *Applied Sciences (Switzerland)*, *12*(6). https://doi.org/10.3390/app12063023

Diaz Rivera, J. J., Akbar, W., Ahmed Khan, T., Muhammad, A., & Song, W. C. (2023). Secure Enrollment Token Delivery Mechanism for Zero Trust Networks Using Blockchain∗. *IEICE Transactions on Communications*, *E106-B*(12), 1293–1301. https://doi.org/10.1587/transcom.2022TMP0005

Diaz Rivera, J. J., Khan, T. A., Akbar, W., Muhammad, A., & Song, W.-C. (n.d.). *Secure enrollment token delivery for Zero Trust networks using blockchain*.

Faísca, J. G., & Rogado, J. Q. (2016). Decentralized semantic identity. *ACM International Conference Proceeding Series*, *13-14-September-2016*, 177–180. https://doi.org/10.1145/2993318.2993348

Hannousse, A., & Yahiouche, S. (2020). *Securing Microservices and Microservice Architectures: A Systematic Mapping Study*. https://doi.org/10.1016/j.cosrev.2021.100415

Ukani, M., Bhadja, S., & Shah, K. (2023). Comprehensive Analysis of Blockchain for Applicability in User Authentication. *2023 2nd International Conference on Futuristic Technologies, INCOFT 2023*. https://doi.org/10.1109/INCOFT60753.2023.10425535

Winder, D. (2025, June 20). Forbes. Retrieved from https://www.forbes.com/sites/daveywinder/2025/06/20/16-billion-apple-facebook-google-passwords-leaked---change-yours-now/