Financial Innovation

RESEARCH Open Access

Sovchain: novel mutual authentication scheme for self-sovereign identity management using blockchain for open banking

Dawood Behbehani[1*], Nikos Komninos[1] and Rajarajan Muttukrishnan[1]

*Correspondence:
dawood.behbehani@citystgeorges.ac.uk

[1] School of Science and Technology, City St George's, University of London, Northampton Square, London EC1V 0HB, UK

Abstract

A significant limitation of Financial-Grade API (FAPI) 2.0 is its inability to verify the true origin of the authenticator used during the authentication process, because it focuses solely on validating the parameters of the request rather than verifying the identity of the sender. To enhance FAPI 2.0 security, we propose SovChain, an approach that embeds machine learning into the pre-authentication phase of self-sovereign identity (SSI) systems, enabling proactive risk assessment before any credential is accepted or validated. By addressing a clear gap in the literature, the absence of a mechanism for early threat detection within SSI, the findings of this study contribute to the development of a more trustworthy open banking ecosystem. SovChain was designed with Hyperledger smart contracts, tested on the TON Internet of Things dataset, and validated using AVISPA for formal security. Simulation experiments evaluated registration and authentication flows, support vector machine classification, communication overhead, and throughput under varying participant loads. SovChain achieved 98.2% accuracy with a 97.7% F1 score. By embedding proactive risk assessment into SSI, SovChain demonstrates the feasibility of combining lightweight machine learning with blockchain-based authentication, offering a scalable and regulation-ready solution for open banking and beyond.

Keywords: Dynamic risk assessment, Open banking, Self-sovereign identity, Support vector machine

Introduction

Open banking relies on FAPI 2.0, an extension of OAuth 2.0 that incorporates stronger security and privacy controls–such as signed requests and encrypted ID tokens–as the standard protocol for secure authorisation. FAPI 2.0 enables users to grant third-party providers limited access to their financial data without sharing login credentials. FAPI 2.0 allows third-party applications to access user resources, typically hosted on a web service or application, without sharing the credentials or sensitive data of users. A framework is provided for secure authorisation and delegated access to protected resources (PRes), such as user data, by offering a standardised method for applications to request and obtain access tokens (ATok). FAPI 2.0 is widely adopted in various

domains, including social media login integration, single sign-on (SSO) solutions, and API access control (Khan et al. 2018). The OAuth 2.0 framework defines several key entities—such as the resource owner, authorisation server, resource server (RS), and client (CLI)—to facilitate secure transactions within IoT networks. These transactions require either pre-established trust or dynamic trust establishment among all involved entities (Dodanduwa 2018). Trust relationships, whether unidirectional or bidirectional, enable access to PRes and underpin the authentication process, during which user identities are verified before system or data access is granted. This process entails authenticating personal identification documents, validating the legitimacy of websites by digital certificates, authenticating historical artefacts, or ensuring the integrity of digital files. Security best practices recommend validating at least two independent factors to ensure robust authentication. Moreover, the FAPI 2.0 security profile is specifically designed for high-value scenarios, serving as a robust security protocol for APIs based on the OAuth 2.0 authorisation framework.

Third-party providers (TPPs) often rely on authentication services such as FAPI 2.0 to manage complex and time-consuming security tasks, including encryption, hashing, token generation, and verification. These services relieve developers from handling lower-level implementations, allowing them to focus on the core functionalities of their applications. However, a significant limitation of the current standard lies in its inability to authenticate the true origin of a request, because it concentrates solely on validating request parameters rather than verifying user identity or device legitimacy. This oversight can lead to unauthorised access to restricted resources due to token theft, inadequate protection mechanisms, or insufficient token validation.

To address identity management and access control in web-based applications, various frameworks and protocols have been developed, including OAuth 2.0, OpenID Connect (OIDC), FIDO2, and WS-Federation. These frameworks are frequently used in conjunction with SSO mechanisms to enable seamless access to multiple services through a single authentication event. Although these technologies have proven effective in several contexts, the rapid advancement of user-centric applications—particularly in the fintech sector, demands more secure and adaptable identity solutions that can address emerging threats and meet growing user expectations.

Novel identity technologies that enhance security, privacy, and user control in fintech applications have garnered significant attention. Self-sovereign identity (SSI) has emerged as a potential solution to meet growing demands, offering a decentralised approach in which users maintain full control over their personal data and digital identities (Ding and Sato 2022; Naik 2020a). SSI has been proposed as a potential solution to overcome the limitations of third-party authentication. SSI is a decentralised approach to digital identification that allows individuals to manage their personal digital identities independently, enabling selective disclosure of personal information when interacting with organisations, services, and other individuals. Traditional identity systems typically grant users limited control over their data, because these systems are governed by centralised authorities such as government agencies, financial institutions, or internet service providers (SPs). Conversely, SSI aims to shift control back to the individual. Blockchain-based SSI offers secure means of conducting online transactions, empowering users to manage their identities and shared information

only when necessary. By allowing selective sharing, SSI plays a critical role in establishing trust in online interactions.

SSI is a decentralised identity management solution that allows individuals greater control over their online personas and personal data. SSI emphasises individual control and ownership over personal identity information. Primarily, SSI allows individuals to manage and distribute their unique features and credentials without requiring a centralised authority or intermediaries. Identity documents are categorised into three types: traditional physical documents (paper/plastic cards), digital identity documents (digital and visual representations of physical documents), and electronic identity documents (digital signature/public key cryptography). These identity credentials require cautious handling, because they are vulnerable to forgery, fraud, and data leakage.

Despite significant progress in SSI research, most studies address authentication reactively, often after credential issuance, leaving pre-authentication risks unresolved. Although FAPI 2.0 is robust in cryptographic protections, does not verify the origin of authenticators, enabling token misuse and impersonation. This paper introduces SovChain, a lightweight blockchain-based mutual authentication framework that embeds a support vector machine (SVM)-driven anomaly detection model into the pre-authentication stage of SSI. SovChain operationalises machine-learning-enabled pre-authentication as a distinct trust layer, thereby filling a neglected gap in SSI research for open banking. SovChain strengthens open banking ecosystems by combining the immutability and auditability of Hyperledger Fabric with real-time risk-aware trust evaluation.

Motivation

Although researchers have applied machine learning (ML) techniques to SSI systems, analysis has remained on isolated components such as biometric spoofing detection, or credential verification. Currently, no comprehensive frameworks are available to enable proactive risk assessment specifically within SSI environments prior to the authentication phase. Existing studies often address risks either reactively or outside the boundaries of the core SSI lifecycle. Consequently, the development of a real-time, ML-driven risk evaluation mechanism embedded in the pre-authentication stage of SSI remains an open and under explored research challenge. Although prior studies (such as Dong et al. 2020; Ahmed et al. 2023) have explored the use of blockchain-based SSI solutions for secure identity management in open banking, their approaches lack mechanisms for detecting and mitigating anomalous behaviour.

Contributions

- This paper proposes a novel approach to embedding ML into the pre-authentication phase of SSI systems, enabling proactive risk assessment prior to the acceptance or validation of any credentials. By addressing a clear gap in the literature—namely, the absence of a holistic, real-time ML framework for early threat detection within SSI—this research contributes towards building a more trustworthy authentication mechanism for open banking.

- A lightweight, blockchain-based mutual authentication scheme is developed for SSI management in open banking systems, integrating the FAPI 2.0 framework for compatibility and secure access control.
- Formal security and empirical validation demonstrating 98.2% accuracy, SAFE verdicts in AVISPA, and superior scalability compared to seven baseline schemes. Collectively, this work addresses a critical gap in SSI by providing an early-warning mechanism against anomalous device and network behaviour, thereby offering a scalable, secure, and regulation-ready authentication protocol for open banking.

Organisation

This paper is organised as follows: Following the introduction presented in Sects. "Introduction", "Related works" provides an overview of related work, and Sect. "Methodology" describes the methodology. Section "Simulation and performance analysis" discusses the simulation and its performance analysis. Section "Discussion" presents the discussion, and Sect. "Conclusion" concludes the paper with highlights for future work.

Related works

Fraud detection

Recent advances in cybersecurity have increasingly explored the relationship between blockchain and ML for secure data handling and anomaly detection. For instance, Su et al. (2021) demonstrated the efficacy of deep learning models in detecting contract level anomalies in decentralised systems. Similarly, Li et al. (2020) applied federated deep learning models for intrusion detection in industrial IoT (IIoT) systems. Existing research in anomaly detection for blockchain and IoT environments has focused on both centralised and decentralised detection architectures. For instance, Golomb et al. (2018) proposed CIoTA, a decentralised anomaly detection model tailored for blockchain-based IoT systems. Broader frameworks combining artificial intelligence (AI) and blockchain, including those discussed by Li et al. (2021), underscore the integration of SVMs and other ML classifiers into decentralised infrastructures. The approach proposed in this study simulates a blockchain-driven network scenario and employs SVM for real-time anomaly detection, offering solutions to existing issues in interpretable yet efficient ML applications within secure blockchain ecosystems. Systematisation of knowledge (SoK), introduced by Hassan et al. (2022) provides an extensive overview of blockchain applications in cybersecurity, directing the focus on secure, decentralised systems. To enable access to digital services, SSI provides a viable alternative to logging in with a username and password provided by an identity provider (IdP). It grants identity subjects authority and ownership of their data. Although an attractive approach, new infrastructures with almost no dependency on the existing infrastructure are necessary (Yildiz et al. 2021). A feasibility study on improving blockchain-based SSI using AI and lightweight cryptography in zero-knowledge proofs detected eight attack vectors (Emati et al. 2023). Specifically, in banking applications, various ML models have been employed for anomaly detection in decentralised environments. Deep learning methods such as long short-term memory (LSTM) networks have demonstrated high accuracy in detecting temporal patterns in fraudulent behavior. Recent studies (Shirley et al. 2024;

Fallah et al. 2024) have integrated ML into blockchains for real-time transactional fraud detection. Cholevas et al. (2024) investigated anomaly detection techniques in blockchain ecosystems from the perspective of unsupervised learning, delving into intricacies and examining cutting edge algorithms to discern deviations from normal patterns. Mienye and Jere (2024) reviewed recent deep learning-based literature and presented a concise description and performance comparison of widely used techniques, including CNN, RNN, LSTM, and GRU, in the context of credit card fraud detection. Bello et al. (2024) proposed a framework that combines ML models with blockchain technology to enhance fraud detection capabilities in decentralised systems, addressing the growing complexity and sophistication of financial fraud schemes. Johnson (2024) explored the effectiveness of different ML algorithms in fraud detection, including logistic regression, random forest, SVMs, gradient boosting, and neural networks, offering comparative performance results on publicly available datasets.

Identity systems

Identity and access management systems allow users to electronically access services and verify ownership through authentication and identity management mechanisms. Digital identities can be classified as:

1. *Centralised Identity Management:*

   - How it works: All user identity data are stored in a single, central database or repository.
   - Advantages: Easy to manage and scale, strong security owing to centralised control.
   - Disadvantages: Can serve as a single point of failure, potentially less user-friendly than decentralised models.

2. *Decentralised Identity Management:*

   - How it works: User identity data are distributed across multiple locations or systems.
   - Advantages: Increased resilience, potentially better user privacy and control.
   - Disadvantages: Can be more complex to manage, may require more sophisticated security measures.

3. *Federated Identity Management:*

   - How it works: Allows users to authenticate and access resources across different domains or systems using a single identity.
   - Advantages: Simplifies user authentication across multiple applications and systems.
   - Disadvantages: Requires agreements and trust relationships between different domains.

4. *User-Centric Identity Management:*

- How it works: Focuses on user identity and their relationship with various resources and systems.
- Advantages: Provides a consistent and personalised user experience.
- Disadvantages: Can be more complex to implement and maintain.

SSI provides a frictionless user experience while prioritising customer privacy and security. Furthermore, it enables rapid credential issuance and verification at any time and place. SSI is vital for successful transactions in an open banking system by eliminating fraudulent activities over the internet. However, SSI faces drawbacks in managing identity over the internet. Thus, this study proposes a novel mutual authentication mechanism for identity management using blockchain technology and SVMs. SVM analyses and classifies traffic into real and fake, preventing fraudulent traffic from entering the open banking system. Blockchain ensures the security of transactions and maintains trust among involved entities.

SSI has emerged as a promising approach to address challenges facing traditional identity systems in financial services. With the growing digital economy, the need for secure and decentralised identity systems has become increasingly crucial, particularly in the financial sector and specifically within open banking environment where customer authentication and authorisation are critical for maintaining trust and compliance. Existing methods such as centralised databases and siloed identity systems have limitations in terms of privacy, control, and portability (Naik 2020b).

Customers must provide personal information manually to each financial institution, leading to inefficiencies and potential security risks. Traditional OAuth-based authentication and authorisation methods present significant security concerns, including phishing attacks, token theft, and session hijacking. Other issues such as insufficient token expiration management, weak client application security, and inadequate scope limitations, further increase risk. The lack of enforced multi-factor authentication and the potential for insecure redirected URIs exacerbate security challenges. Moreover, trust issues with third-party applications and the complexities of proper OAuth implementation result in misconfigurations, heightening the overall security concerns.

SSI offers a solution to aforementioned challenges by granting individuals control over their digital identities and enabling secure, decentralised authentication and authorisation (McLaughlin et al. 2009). By leveraging blockchain technology and cryptographic techniques, SSI systems enable users to manage their own identity data, determine the information to be shared, and maintain control over their personal information (Naik 2020b). Compared to conventional identity approaches, SSI provides several benefits such as enhanced privacy, increased user control, and improved security through the use of decentralised identifiers and verifiable credentials (Liu et al. 2020).

Recent studies have explored various aspects of SSI considering financial services, including the development of secure and decentralised authentication protocols, the design of blockchain-based identity management systems, and the application of SSI principles to address the unique requirements of the financial sector (Ma et al. 2022; Liu et al. 2020).

A study proposed a blockchain-based system for credit banking and massive online open courses using a novel heterogeneous multi-chain blockchain network structure

based on the Polkadot framework (Qu 2021). Another study introduced a secure and decentralised SSI authentication protocol that incorporated privacy protection and fine-grained access control mechanisms (Ma et al. 2022). This protocol leveraged the blockchain technology, decentralised identifiers, and verifiable credentials to enable users to authenticate themselves and control their access to personal information. Research has focused on the design patterns of blockchain-based SSI systems, and emphasised key considerations in areas such as key management, decentralised identifier management, and credential design (Liu et al. 2020).

The approach proposed in this paper differs from those of previous studies; investigations focus on detecting fraudulent activities at the initial stage, before authentication, thereby preventing the perpetrator from executing unauthorised transactions. The proposed approach uses SVMs for their efficiency, lower computational cost, and explainability. Although simple, the model achieves competitive accuracy (98.2%) on the TON_IoT dataset when deployed within a blockchain-based simulation framework. Table 1 presents a comparative analysis of the proposed scheme with a other related works.

Recent studies in blockchain-based SSI models have advanced privacy preservation and decentralization, yet they offer limited mechanisms for proactive, risk-aware trust evaluation before credential issuance or authentication (Kulabukhova 2020; Putra 2025; Ding 2023). This gap exposes open-banking ecosystems to credential misuse and identity fraud, because existing SSI frameworks primarily implement reactive verification once compromise occurs. SovChain directly addresses this gap by embedding machine-learning-enabled pre-authentication anomaly detection within the SSI trust establishment process. Grounded in the zero-trust architecture (ZTA) paradigm (Stafford 2020), SovChain operationalizes a proactive security model in which every entity and credential request is continuously verified and risk-scored prior to interaction. This approach shifts the theoretical foundation of SSI from static credential trust to dynamic, evidence-based trust formation, aligning with emerging directions in risk-based digital identity management (Naik 2020a). Thus, SovChain not only strengthens the theoretical linkage between trust, verification, and decentralization, but also provides an implementable framework that fuses blockchain immutability with adaptive ML-driven risk assessment, extending SSI research into the proactive security domain most relevant to modern open-banking compliance requirements.

To conclude, despite the growing advantage of ML in enhancing security, current research underscores several critical gaps in its application to SSI systems, particularly for proactive risk assessment before authentication. One notable drawback is the absence of comprehensive, end-to-end frameworks that incorporate ML-driven risk evaluation across the entire SSI lifecycle. Much of the existing work remains fragmented, often focusing on narrow aspects such as biometric spoofing or document forgery, without integrating these components into a cohesive architecture.

Methodology

System design

SovChain is an SVM-enabled novel mutual authentication mechanism for open banking systems that is integrated with FAPI 2.0. SovChain integrates three tightly

Table 1 Comparative table between the proposed model against other models in the literature

Method/study	Dataset used	ML model	Approach type	Blockchain integration	Notes
LSTM-based approach enhancing fraud detection (Cruzpero 2024)	Ethereum fraud detection dataset	LSTM with attention	Reactive	Yes	Captures temporal dependencies in ethereum transactions
Deep autoencoder ensembles for blockchain anomaly detection on blockchain (Scicchitano et al. 2020)	Custom blockchain dataset	Deep Autoencoder	Reactive	Yes	Focuses on non-black box graph techniques for anomaly detection
Optimizing anomaly detection through machine learning: a comparative analysis for IoT datasets (Balega et al. 2024)	TON_IoT, IoT-23, NSL-KDD	XGBoost	Reactive	No	Demonstrates high accuracy but lacks blockchain integration
Credit card fraud detection using autoencoder model in unbalanced datasets (Zou et al. 2019)	Credit card transactions dataset	Autoencoder	Reactive	No	Addresses fraud detection in unbalanced datasets
Secure sentinel leveraging machine learning for fraud detection in blockchain transactions (Shirley et al. 2024)	Ethereum fraud detection dataset	Secure sentinel	Reactive	Yes	Addresses fraud detection through transaction such as credit card, etc
SovChain	Device/network traffic	TON_IOT dataset	Proactive	Yes	Addresses fraud detection through anomalous device behaviour

coupled components: a blockchain-based mutual-authentication layer (implemented on Hyperledger Fabric), a pre-authentication risk-evaluation layer powered by an SVM, and smart contracts for ledgered audit and access control. SovChain employs a novel mutual authentication scheme in a Hyperledger blockchain environment, a permissioned blockchain with a membership infrastructure that enables network participants to both strongly authenticate and validate permission to perform diverse system processes, such as reconfiguration. With the assistance of blockchain-based Hyperledger smart-contract tokens, the proposed system enables customers to allow third parties of financial institutions access to their server resources without disclosing sensitive credentials.

Before successful authentication, the system checks for anomalous device behaviour by quantifying the risks using an SVM, following a supervised learning method. SVMs are particularly effective in detecting abnormal behaviour in fraudulent transactions owing to their ability to handle high-dimensional data and complex nonlinear

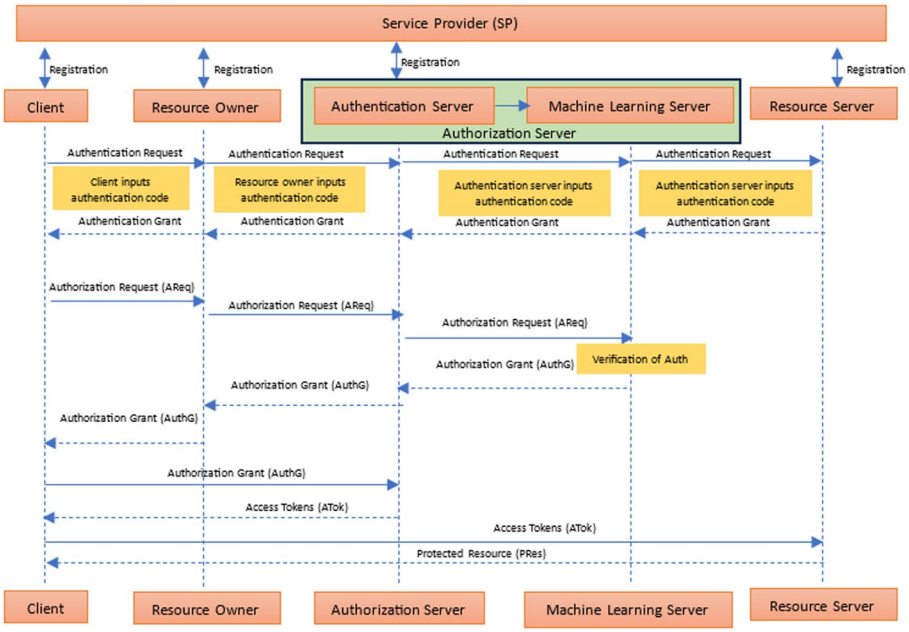


Fig. 1 SovChain sequence diagram

relationships. Additionally, the flexibility of SVMs in both binary and multi-class classification, along with their proven performance in real-world applications, makes them an excellent selection for accurately identifying fraudulent activities (Rtayli and Enneya 2020).

Notably, SovChain embeds a mutual authentication protocol within a permissioned blockchain (Hyperledger) environment that provides secure, bidirectional trust between transaction parties. Accordingly, SVM-based supervised learning models device behaviour, solving a convex optimisation problem to classify interactions as legitimate or anomalous in real time. This integration of formal mutual authentication with grounded anomaly detection within an open banking framework addresses critical issues concerning trust, privacy, and security in decentralised financial ecosystems.

The entities involved in the SovChain system include a Client (CLI), resource owner, authentication server (AS), ML server, and resource server, as shown in Fig. 1. All entities must be registered with the blockchain network prior to authorisation. The authorisation steps are as follows:

1. All the entities register with the service provider (SP) per the algorithm discussed in Sect. "Authentication algorithm"
2. Based on the FAPI 2.0 framework, all the entities must authenticate themselves with the other entities with whom they communicate per the algorithm discussed in Sect. "Authentication algorithm".
3. CLI initiates the authorisation process in blockchain Hyperledger by sending an AReq to the resource owner (as outlined in the algorithm in Sect. "Authorisation and

access token generation for protected resource transfer"), which sends an acknowledgement.

4. The authorisation server sends the AReq to the ML server for fraud detection. Upon successfully identifying the legitimacy of the CLI through the SVM classification algorithm, the ML server sends the Authorisation Grant Message (AuthG) to the resource owner (as outlined in algorithm in Sect. "Authorisation and access token generation for protected resource transfer").
5. The resource owner further authenticates the CLI by the mutual authentication algorithm (Sect. "Authorisation and access token generation for protected resource transfer"), which sends the AuthG.
6. The CLI employs the AuthG to obtain an authorisation token (ATok) from the authorisation server (as discussed in the algorithm in Sect. "Authorisation and access token generation for protected resource transfer").
7. In the final phase, the CLI transmits ATok to RS (RS) to obtain the protected resource (PRes) (as discussed in algorithm in Sect. "Authorisation and access token generation for protected resource transfer").

SovChain's workflow has three major phases: (1) registration, in which entities register their permanent identifiers and secret material on the permissioned ledger; (2) pre-authentication risk assessment, in which, on each incoming authorisation request, the AS forwards a lightweight feature vector to the ML server, where the SVM classifies the request as normal or anomalous; (3) mutual authentication and token issuance, in which, if the SVM result is benign, the mutual authentication protocol runs (using Hyperledger smart contract-anchored values), and the AS issues an access token (ATok) that the client uses to access PRes on the RS.

For building authorisation and trust (authentication), each pair of entities should follow a mutual authentication scheme. The authorisation server comprises an AS and an ML server integrated as a single entity to reduce complexity (if required).

Blockchain-based mutual authentication scheme

Two phases of authorisation and subsequent trust building exist between different pairs of entities of SovChain for SSI management:

1. The registration phase specifies the registration of an entity in a peer-to-peer network (blockchain), also known as an SP. All entities should follow the same procedure during the registration phase.
2. The authentication phase is executed when a resource must be transferred between a pair of entities. For instance, when the RS delivers a PRes to the CLI, the authentication state confirms the authenticity of the RS and the CLI.

The registration of entities occurs only once in a secure channel. However, the authentication process occurs multiple times over an insecure channel whenever resources are to be transferred. Table 2 lists the abbreviations and descriptions used in these algorithms, and Table 3 lists the abbreviations used for SovChain algorithms.

Table 2 Mathematical notations and cryptographic operators used in the SovChain

Symbol	Description
$A \bigoplus B$	Exclusive-or (XOR) operation between two strings A and B of the same length
A ‖ B	Concatenation operation between two strings A and B
H()	One-way hash function
μ	Mean (used in normalisation process)
σ	Standard deviation (used in normalisation process)

Table 3 Abbreviations used for SovChain algorithms

Symbol	Description
CLI	Client
RO	Resource owner
RS	Resource server
AS	Authentication server
SP	Service provider
PRES	Identity of PRES
sk*X*	Secret key of an entity X
pw*X*	Password of an entity X
id*X*	Identity of an entity X
skRA	Session key generated by RS and AS
skSP	Secret key of SP
m, d, s, s , Nm, Nd, Nd1	Random numbers
$A \bigoplus B$	Exclusive-or operation between A and B
A ‖ B	Concatenation operation between A and B
H()	One-way hash function
chain 1	Stores local data in the blockchain
chain 2	Stores shared data in the blockchain

Registration algorithm

For registration, each entity (each CLI, resource owner, AS, ML server, and RS) is registered with the peer-to-peer network (blockchain), that is, the SP. The protocol flow for the registration of a CLI under the SP, is shown in Protocol Flow 1.

Algorithm 1 Protocol Flow 1: Client (*CLI*) Registration

Input: id_{CLI}, pw_{CLI}
Output: Stored credentials in Chain1 and Chain2
At Client CLI**:**

1. Choose random nonce m.
2. Compute:

$$ACLI = H(id_{CLI} \oplus pw_{CLI} \oplus m), \quad PCLI = H(id_{CLI}).$$

3. Send $(pid_{CLI}, PCLI)$ to SP over Chain2.

At Service Provider SP**:**

1. Receive $(pid_{CLI}, PCLI)$.
2. Choose random nonce s and compute:

$$BCLI = pid_{CLI} \oplus s,$$

$$CCLI = H(pid_{CLI} \| s), \quad DCLI = H(s \| pid_{CLI}),$$
$$sk_{CLI} = H(PCLI \oplus sk_{BC}),$$
$$PSCLI = PCLI \oplus H(sk_{BC} \| id_{BC}).$$

3. Store $(pid_{CLI}, s, CCLI)$ in Chain2 and $(PSCLI, sk_{CLI})$ in Chain1.
4. Send $(BCLI, CCLI, DCLI, sk_{CLI}, s)$ to CLI over Chain2.

At Client CLI**:**

1. Receive $(BCLI, CCLI, DCLI, sk_{CLI}, s)$.
2. Compute:

$$ECLI = BCLI \oplus H(pw_{CLI}),$$
$$FCLI = CCLI \oplus H(pw_{CLI}),$$
$$GCLI = DCLI \oplus H(pw_{CLI}),$$
$$tsk_{CLI} = sk_{CLI} \oplus H(pw_{CLI}).$$

3. Store $(ACLI, ECLI, FCLI, GCLI, tsk_{CLI}, m, s)$ in Chain1.

The following procedures are followed when a resource owner considers registration under an SP (Registration Phase):

Algorithm 2 Registration Phase for Resource Owner (*RO*)

Input: id_{RO}, pw_{RO}
Output: Stored credentials in Chain1 and Chain2
At Resource Owner RO**:**

1. Choose random nonce d.
2. Compute:
$$ARO = H(id_{RO} \oplus pw_{RO}) \oplus d,$$
$$pid_{RO} = H(id_{RO} \oplus d), \quad PRO = H(id_{RO} \| d).$$
3. Send (pid_{RO}, PRO) to SP over Chain2.

At Service Provider SP**:**

1. Receive (pid_{RO}, PRO).
2. Choose random nonce s and compute:
$$BRO = pid_{RO} \oplus s,$$
$$CRO = H(pid_{RO} \| s), \quad DRO = H(s \| pid_{RO}),$$
$$sk_{RO} = H(PRO \oplus sk_{BC}),$$
$$PSRO = PRO \oplus H(sk_{RO} \| id_{RO}).$$
3. Store (pid_{RO}, s, CRO) in Chain2 and $(PSRO, sk_{RO})$ in Chain1.
4. Send $(BRO, CRO, DRO, sk_{RO}, s)$ to RO as an acknowledgment over Chain2.

At Resource Owner RO**:**

1. Receive $(BRO, CRO, DRO, sk_{RO}, s)$.
2. Compute:
$$ERO = BRO \oplus H(pw_{RO}),$$
$$FRO = CRO \oplus H(pw_{RO}),$$
$$GRO = DRO \oplus H(pw_{RO}),$$
$$tsk_{RO} = sk_{RO} \oplus H(pw_{RO}).$$
3. Store $(ARO, ERO, FRO, GRO, tsk_{RO}, d, s)$ in Chain1.

Authentication algorithm

In the authentication phase, when a resource must be transferred between entities, Authentication Flow is used. Assuming that an RS wishes to transfer a PRes to an AS. The RS and AS authenticate each other and the PRes and then confirm the transfer. The authentication protocol steps employed among the RS, AS, and SP to handle the transfer of PRes from the RS to the AS are explained further. Because the AS is registered under the SP, it can be assumed that the SP has stored (PSAS, skAS) in blockchain chain 1 similar to (PSRS, skRS) of the RS. The authentication phase is presented in authentication flow and is depicted schematically in Fig. 2.

Algorithm 3 Authentication Protocol between *RS*, *AS*, and *SP*

Input: $idRS, pwRS, PresID$
Output: Session key sk_{RA}
At Resource Server RS**:**

1. Compute $ARS' = H(idRS \oplus pwRS) \oplus m$ and verify $ARS' = ARS$.
2. Retrieve $(ARS, ERS, FRS, GRS, tskRS, m, s)$ from Chain1.
3. Compute:

$$pidRS = H(idRS \oplus m),\ BRS = ERS \oplus H(pwRS),\ CRS = FRS \oplus H(pwRS),\ DRS = GRS \oplus H(pwRS).$$

4. Choose nonce N_m and compute:

$$CRS' = CRS \oplus s,\ PresRS = PresID \oplus s,\ pidRS' = pidRS \oplus N_m.$$

5. Send $(CRS', N_m, pidRS', PresRS)$ to AS.

At Authentication Server AS**:**

1. Receive $(CRS', N_m, pidRS', PresRS)$.
2. Retrieve $(AAS, EAS, FAS, GAS, tskAS, d, s)$ from Chain1.
3. Choose nonce N_d, compute:

$$sk_{AS} = tskAS \oplus H(pwAS),\ QAS = H(N_d \| sk_{AS}),$$

$$PAS = H(idAS \| d),\ PAS' = PAS \oplus N_d.$$

4. Send (QAS, PAS', N_d) to SP.

At Service Provider SP**:**

1. Compute $PAS = PAS' \oplus N_d$.
2. Compute $PSAS = PAS \oplus H(skSP \| idSP)$.
3. Retrieve sk_{AS} and verify $QAS = H(N_d \| sk_{AS})$.
4. If valid, authenticate AS and release $(pidRS, s, CRS)$ to AS.

Back at AS**:**

1. Recover $pidRS = pidRS' \oplus N_m$.
2. Verify $CRS = CRS' \oplus s$ and $PresID = PresRS \oplus s$.
3. Generate s', N_{d1} and update chain values.
4. Compute session key:

$$sk_{RA} = H(DRS \| N_{d1} \| N_m).$$

5. Send $(RRS, DRS_{new}, CRS_{new}, BRS_{new}, s'_{new}, N_{d1})$ to RS.

At RS**:**

1. Verify $DRS = RRS \oplus BRS$.
2. Derive updated secrets and compute:

$$sk_{RA} = H(DRS \| N_{d1} \| N_m).$$

3. Update stored values in Chain1.

15

For the remainder of the chain, the same sequence of actions is followed between the AS and the resource owner, and then between the resource owner and the CLI.

Authorisation and access token generation for protected resource transfer

After successful authentication between the entities, an authorisation request (AReq) is sent to the resource owner by the client. The latter resends it to the authorisation server and subsequently to the ML server to check for fraudulent behavior during the course of action. If the request is cleared, the ML server sends an AuthG to the AS and subsequently to the CLI via the resource owner, as shown in Fig. 1. Upon receipt of the AuthG, the CLI resends to the authorisation server, which matches the two AuthG, and

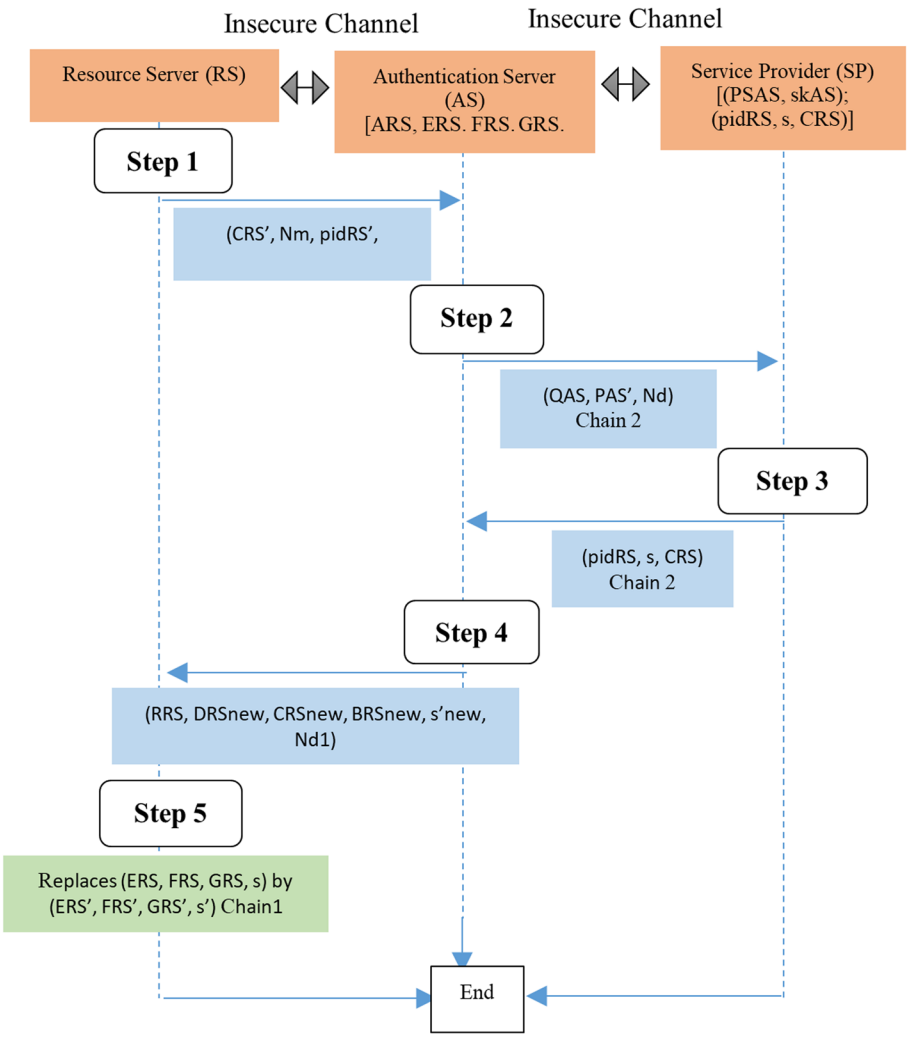


Fig. 2 Authentication between RS, AS, and SP

upon a successful match, it sends an ATok to the CLI. The CLI sends the ATok to the RS, which then sends a PRes to the CLI.

SVM-enabled ML server

The SVM-enabled ML server checks for anomalous device behaviour. The CLI network traffic data are analysed in this module and classified into attack/non-attack vectors using the TON_IOT dataset, which is used to train, validate, and test the proposed model for network attack classification.

The SVM model can be formulated as follows:

- Given a training set $\{(x_i, y_i)\}_{i=1}^{N}$ where $x_i \in R^n$ are feature vectors (e.g., device behavior statistics, transaction attributes) and $y_i \in \{-1, +1\}$ are class labels (normal or anomalous).
- SVM finds a hyperplane defined by $\mathbf{w} \cdot \mathbf{x} + b = 0$ that maximally separates the two classes with the largest margin.

- Formally, the optimization problem can be expressed as:

$$\min_{w,b} \frac{1}{2}\|w\|^2 \tag{1}$$

subjected to the constraints

$$y_i(w \cdot x_i + b) \geq 1, \quad \forall i \tag{2}$$

- For non-linearly separable data, SVM uses kernel functions $K(x_i, x_j)$ to implicitly map data into a higher-dimensional feature space where it is separable.
- The SVM decision function can be expressed as:

$$f(x) = \text{sign}\left(\sum_{i=1}^{N} \alpha_i \, y_i \, K(x_i, \, x) + b\right) \tag{3}$$

where α_i denotes learned weights for the support vectors.

- The features x represent device behavior metrics, transaction parameters, and user activity profiles. By solving SVM optimisation, SovChain classifies new transactions or device requests as normal or anomalous, enabling real-time anomaly detection.

Dataset

The TON_IoT dataset is widely used in cybersecurity research for evaluating the performance of AI-based intrusion detection systems in IoT environments. These datasets contain telemetry, network traffic, and system log data from IoT and IIoT devices. Notably, IIoT data differ significantly from consumer IoT data generated by devices such as smart speakers (e.g., Alexa), smart lights, or home automation hubs. Consumer IoT environments focus on usability and convenience, whereas Industry 4.0 datasets reflect high-stake, mission-critical environments where failures or breaches impact operational continuity and physical safety. Therefore, using TON_IoT data enables the evaluation of the proposed system in a realistic, high-risk setting that better aligns with enterprise-level applications such as open banking and secure identity frameworks

System model

The architecture of the proposed model comprises three major sections: Section I includes the data preprocessing modules, Section II presents the classification tasks for several network attack vector classifications, and Section III includes the classified attack vectors as outputs as outlined in Table 4.

Data preprocessing is necessary to reduce noise in the dataset. Initially, the dataset is converted into a numerical form, followed by standardisation and normalisation. Feature scaling is one of the most important data preprocessing procedures in ML. When data are not scaled, algorithms that calculate the distance between features turn biased toward numerically greater values. Standardisation and normalisation are two common techniques for feature scaling. Feature scaling transforms the values of features or variables in a dataset to a similar scale, allowing for the easy construction of accurate and

Table 4 Comparative performance analysis

Feature	Description
ts	Timestamp of connection between flow identifiers
src_ip	Source IP addresses
src_port	Source ports
dst_ip	Destination IP addresses
dst_port	Destination ports
proto	Transport layer protocols of flow connection
service	Dynamically detected protocols
duration	Timing of the packet connection
src_bytes	Source bytes from TCP sequence numbers
dst_bytes	Destination bytes from TCP sequence numbers
conn_state	Various connection states
missed_bytes	Number of missing bytes in content gaps
src_pkts	Number of original source IP packets
src_ip_bytes	Number of original IP bytes
dst_pkts	Number of destination packets
dst_ip_bytes	Number of destination IP bytes
http_trans_depth	Pipelined depth into the HTTP connection
http_method	HTTP request methods
http_url	URLs used in the HTTP request
http_version	HTTP version utilised
http_request_body_len	Actual uncompressed content sizes of the data transferred from the HTTP CLI
http-response_body_len	Actual uncompressed content sizes of the data transferred from the HTTP server
http_status_code	Status codes returned by the HTTP server
http_user_agent	Values of the user agent header in the HTTP protocol
http_orig_mime_types	Ordered vectors of mime types from source system in the HTTP protocol
http_resp_mime_types	Ordered vectors of mime types from destination system in the HTTP protocol
dns_query	Domain name subjects of the DNS queries
dns_qclass	Value that specifies the DNDS query classes
dns_qtype	Value that specifies the DNDS query types
dns_rcode	Response code values in the DNS responses
dns_AA	Authoritative answers of DNS
dns_RD	Recursion desired of DNS
dns_RA	Recursion available of DNS
dns_rejected	DNS queries are rejected by the server
ssl_version	SSL version that is offered by the server
ssl_cipher	SSL cipher suite that the server chose
ssl_resumed	SSL flag indicating the session that can be used to initiate new connection
ssl_established	SSL flag indicates establishing connection between two parties
ssl_subject	Subject of the X.509 cert offered
ssl_issuer	Trusted owner of SSL and digital certificate
weird_name	Names of violations related to protocols
weird_addl	Additional information associated to protocol violations by the server
weird_notice	Indicates if the violation was turned into a notice
Label	Tag normal and attack records
Type	Tag attack categories

effective ML models. The process of creating features that appear to be roughly normally distributed is known as standardisation. The process moves values such that they are placed about the mean, with the mean set to zero, and the rescaled data distribution has a unit standard deviation. Standardisation is calculated as follows by:

$$x\text{new} = (x - \mu) \ / \ \sigma \tag{4}$$

where x denotes the original feature vector, μ denotes the mean of the feature vector, and σ denotes the standard deviation. Normalisation is the process of shifting and rescaling data; therefore, the data range remains between [0,1]. Normalisation is calculated as follows:

$$x\text{new} = (x - x\min)/(x\max - x\min) \tag{5}$$

where x denotes the original feature vector, $xmin$ denotes the minimum value of the feature vector, and $xmax$ denotes the maximum value. Both standardisation and normalisation are critical in ML, particularly for SVM, depending on the type of dataset used. The selection of standardisation and normalisation depends on feature distribution and range. Normalisation is useful when features are of different scales or when no outliers exist. Standardisation is useful when features have a normal or Gaussian distribution, or when outliers exist. In the proposed model, both techniques are applied simultaneously to achieve best results. Both standardisation and normalisation are applied in a structured, feature-dependent manner. Specifically, continuous-valued features such as CPU usage, memory load, and packet sizes are standardised using z-score transformation to centre them around zero and reduce the impact of outliers. Categorical and bounded-range features including binary protocol indicators and flags are normalised to the [0,1] range using minmax scaling. In cases where standardisation improves feature variance but the overall scale remains inconsistent across features, the standardised values are normalised to ensure consistency for distance-based metrics in SVM. This two-step preprocessing strategy has been empirically found to improve SVM kernel performance and convergence stability. In the classification section, the preprocessed dataset is classified into two parts: 70% of the dataset is used for training the model, whereas 30% is used for testing and validation using SVM. Thus, different types of attacks are classified.

Simulation and performance analysis

Simulation environment

Blockchain-based mutual authentication scheme

The proposed technique was tested on a system equipped with a 2.30 GHz Intel i3 processor and 10 GB of RAM running Ubuntu 18.04.6 LTS. The simulation environment utilised a Hyperledger fabric network, with Hyperledger composer operating within the Hyperledger playground environment. A composer was employed to define the business network, including Model (.cto), Script (.js), and Access Control (.acl) files. These components were deployed locally on localhost8080/ as an archive, as illustrated in Fig. 3. During implementation, multiple participants were created, including the CLI, resource owner, AS, and RS, each with a unique identity managed at localhost:8080/identity. All participants were registered with a single SP. Following registration, authentication between the participants was executed successfully. Several sample runs were performed

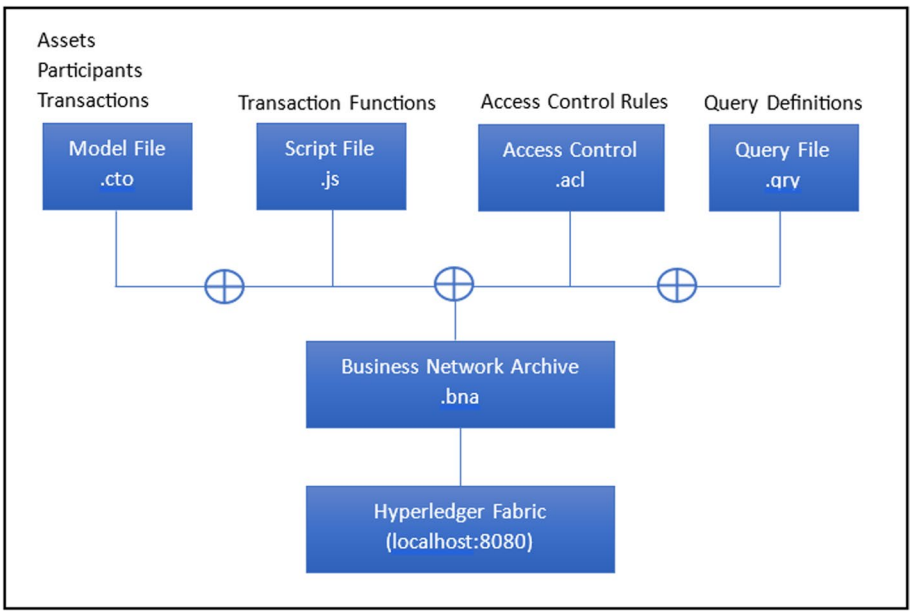


Fig. 3 Business network definition components in composer

to evaluate the registration and authentication states in a pairwise manner: between the CLI and resource owner, resource owner and AS, AS and RS, and CLI and RS. In these test runs, both successful and unsuccessful authentication attempts were examined via localhost:8080/test, and the resulting assets were validated. No anomalies were detected during the implementation, indicating the robustness of the scheme.

SVM-enabled ML server

The TON_IoT dataset for network traffic was initially pre-processed using Python, which involved converting the data to numerical values, handling missing entries, removing null values, and categorising categorical data into predefined groups. Following the preprocessing and data cleaning stages, the dataset was normalised to constrain the values of all columns to a range between 0 and 1, inclusive. Thereafter, the dataset was prepared for EDA, as outlined in Rao et al. (2021). EDA, derived on the basis of methods described by Szoka (1982), included various plots to facilitate a comprehensive analysis of the data, as follows

1. Bar plots were used to assess the correlation between the label (normal or attack) and feature vectors, error distribution and concentration of values.
2. Kernel density estimation plots were applied for non-parametric probability density estimation of feature vectors.
3. Box plots illustrated outliers, spread, and skewness in the data.
4. A heat map (correlation matrix) was employed for feature extraction, highlighting relevant features with significant correlation values and excluding those with minimal correlation.

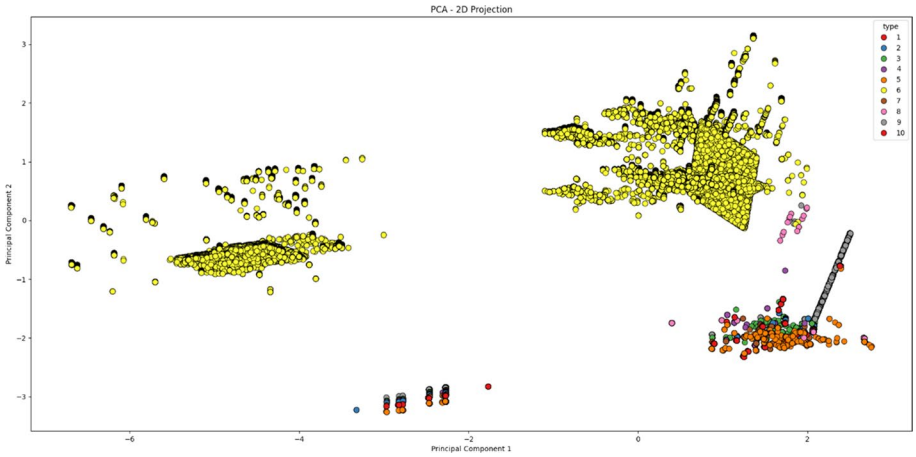


Fig. 4 Two-dimensional PCA projection showing the separability of normal and attack traffic classes

Although initial exploratory data analysis (EDA)—including correlation heatmaps and time series plots—assisted in understanding feature relationships and temporal trends, the final selection of classifier is determined through empirical performance evaluation rather than assumptions on linear separability. To assess the separability of classes in feature space, principal component analysis (PCA) is employed. As shown in Figure 4, the classes exhibit clear clustering patterns in the first two principal components, suggesting that a margin-based classifier such as SVM is appropriate for this task.

The PCA plot provides a linear projection of the high-dimensional dataset into two dimensions. Despite the inherent dimensionality reduction, several distinct clusters are observed, indicating underlying class separability. Classes such as 2, 3, 6, and 9 appear to occupy relatively discrete regions of the projected feature space, suggesting that the original data possessed a degree of linear or near-linear separability. Although some class overlap is observed—particularly involving the dominant class (class 5)—the visible structure implies that a margin-based classifier such as a SVM can serve as effective. SVM is well-suited to this setting owing to its ability to find a hyperplane that maximizes the margin between classes in the original high-dimensional space. Even if the data are not perfectly separable in two dimensions, the PCA visualisation supports the hypothesis that SVM can construct robust decision boundaries in the full feature space. Additionally, the presence of some class imbalance, as suggested by the spread and density of class 5, can be mitigated by employing class weighting within the SVM framework.

Experimental procedure

- Deploy the Hyperledger network locally and register entities (Phase 1: 40 participants; Phase 2: 75 participants).
- Execute registration and authentication workflows and measure blockchain metrics (execution time, latency, throughput, asset operations). Repeat for multiple runs to compute averages.

- For ML: Split the TON IoT dataset into training (70%) and test (30%) sets; run feature preprocessing; train SVM with grid-search and five-fold cross validation; evaluate using the test set; compute metrics and the confusion matrix.
- Formal verification: Encode the protocol in high-level protocol simulation language (HLPSL); run AVISPA back-ends, including on-the-fly model-checker (OFMC) and CL-based attack search engine (CLAtSe), and record results (search time, visited nodes, SAFE/UNSAFE).
- Comparative evaluation: measure and tabulate communication overhead (bits), execution time per protocol, and supported security functions versus seven baseline protocols (as shown in Table 5).
- Conduct robustness checks (vary kernel, different splits, k-fold CV) and report stability.

Simulation results and performance analysis

Two distinct datasets were used to assess the performance of the scheme. The first dataset, used in Phase 1 of the experiment, comprised one SP, two resource owners, four ASs, six RSs, and 27 CLIs, and the second, used in Phase 2 of the experiment, comprised one SP, three resource owners, three ASs, six RSs, and 62 CLIs. The performance of the scheme was evaluated using three parameters: total execution time, average latency, and throughput of the transactions held in the two datasets. Evaluating the performance of the scheme using these three parameters was essential because it provided a comprehensive assessment of system efficiency, responsiveness, and ability to handle high transaction volumes. The total execution time measures how swiftly tasks were completed, average latency assesses the promptness of individual transaction processing, and throughput indicates the capacity of the system to manage large workloads. These parameters ensured that the scheme satisfied the necessary performance standards and could be scaled effectively under varying demands.

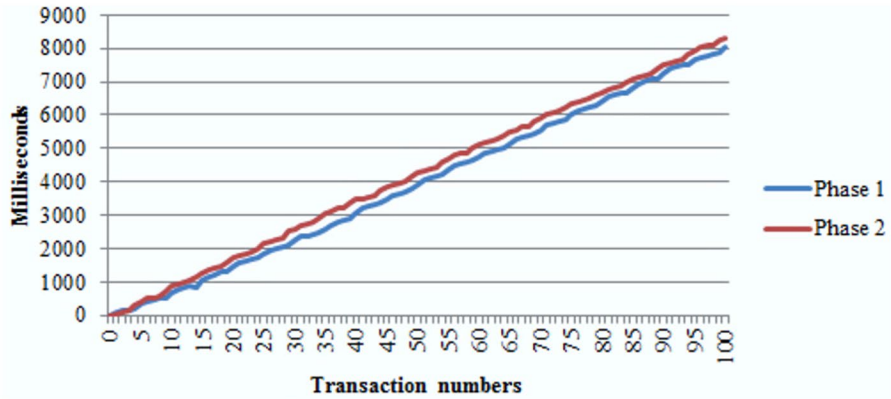
The three parameters were computed as follows:

- Total execution time: Total execution time denotes the total time taken by transactions in milliseconds calculated by:

Table 5 Comparative performance analysis

Scheme	Communication overhead (Bits)	Execution time (ms)	Supported SFs	No. of SFs
Yu et al. (2020)	4000	$22T_H = 11$	SF (1, 9, 10, 11, 12, 13)	6
Lee, C.- C., Lai, Y.- M., Chen, C.- T., Chen, S.- D. (2016)	3872	$28T_H = 14$	SF (1, 4, 5, 12)	4
Rahmani et al. (2021)	3232	$18T_H = 9$	SF (1, 2, 4, 5, 9, 10, 11, 12)	8
Hasan et al. (2022b)	4032	$12T_{ECCM} + 12T_H + 2T_{ASK} = 1806.9$	SF (2, 5, 7, 8, 12)	5
Dwivedi et al. (2020b)	6688	$8T_H + 7T_{ASK} = 3658$	SF (2, 5, 7, 8, 12)	5
Palit et al. (2022)	5584	$16T_H + 2T_{SK} = 25.4$	SF (1, 2, 3, 10)	4
SovChain	3232	$20T_H = 10$	SF (1–13)	13

(a) Registration Phase



(b) Authentication Phase

Fig. 5 Total execution time for the registration and authentication phases in the blockchain network

$$\sum_{i=1}^{n} Ti, \tag{6}$$

where Ti denotes the execution time of the ith transaction and n denotes the overall number of transactions. Ti is measured by computing the difference between the transaction completion time and the transaction submission time in the blockchain Fig. 5.

- Average latency: Average latency is obtained by:

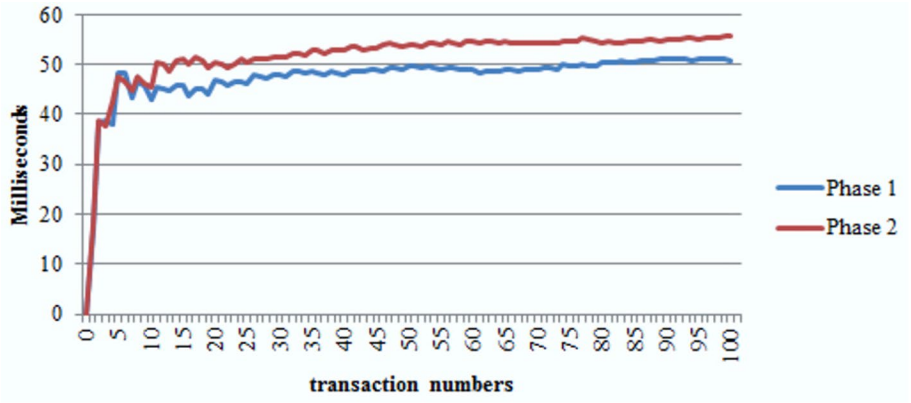
$$\frac{\sum_{i=1}^{n} Ti}{n} \tag{7}$$

measured as the average of the total execution time in milliseconds for a group of transactions Fig. 6.

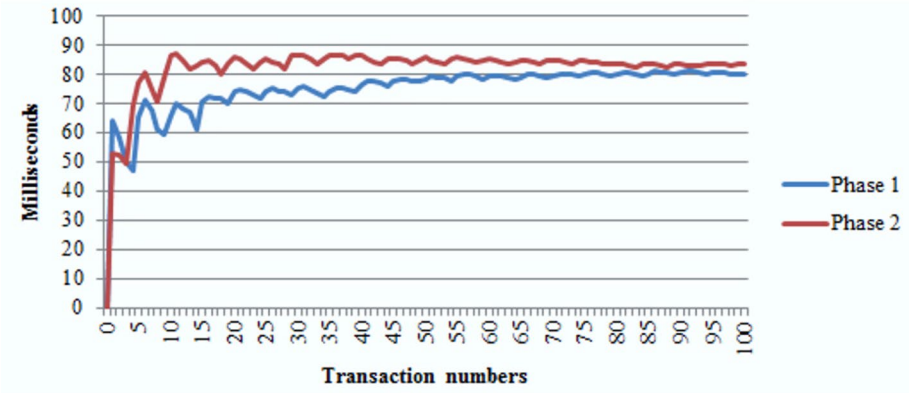
- Throughput: The total number of transactions executed per second is calculated by:

$$\frac{\frac{1000}{\sum_{i=1}^{n} Ti}}{n} \tag{8}$$

Figures 8 and 9 present the asset creation and read times, respectively. Fast creation time indicates the registration of new users or credentials quickly, which is critical for



(a) Registration Phase



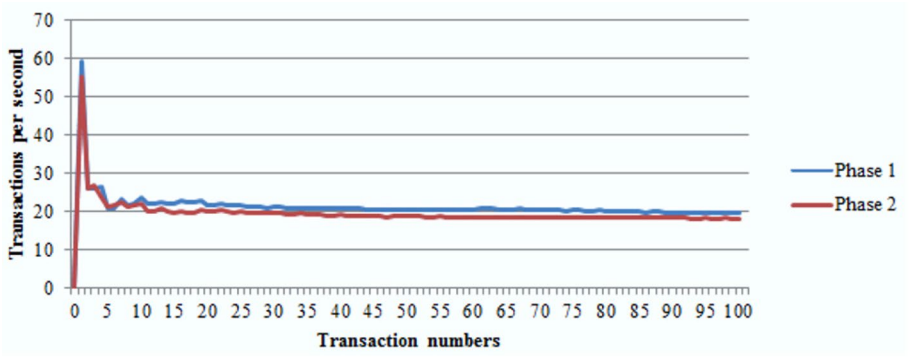
(b) Authentication Phase

Fig. 6 Average transaction latency for the registration and authentication phases under varying participant loads

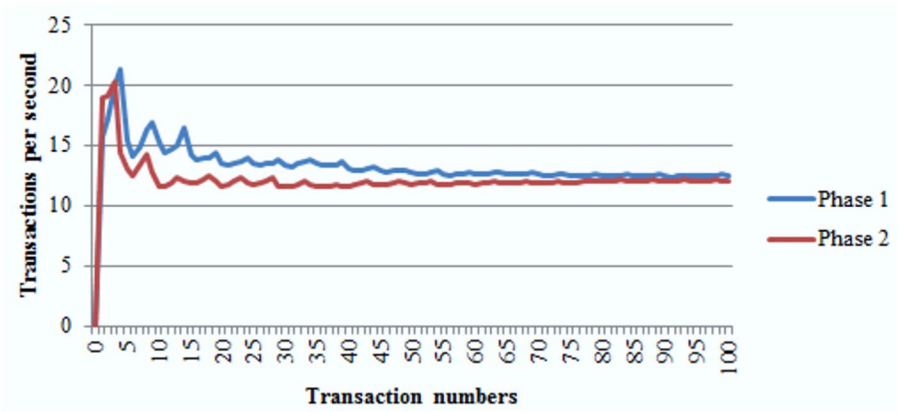
scalability in open banking environments with frequent user onboarding. An increase in creation times with user volume indicates performance bottlenecks in smart-contract execution or consensus protocols. Read times indicate how quickly identity or authentication data are retrieved from the ledger.

The process is essential for real-time decision-making, including verifying credentials during login or transaction approval. In financial systems, delays in read time negatively impact user experience and system responsiveness.

An analysis of the extracted feature vectors revealed some important characteristics as listed in Table 4. Characteristics of the features, such as magnitude (0), magnitude (1), data type, and reliability, were obtained from the bar plots. The magnitudes were depicted directly in bar plots. The data type was balanced or unbalanced by checking the relative height of the bars. When bar heights for the target values of both 0 and 1 were equal or nearly equal (equality $\geq 60\%$), we considered it as a balanced data type; otherwise, it was considered imbalanced. The reliability was indicated by black lines in the middle of the bars. These were referred to as error bars. Short error bars or the absence of an error bar indicated reliable data, whereas long error bars indicated unreliable data. Outliers, spread (0), and spread (1), were obtained directly from the box plots. Box plots



(a) Registration Phase



(b) Authentication Phase

Fig. 7 Transaction throughput for the registration and authentication phases under varying participant loads

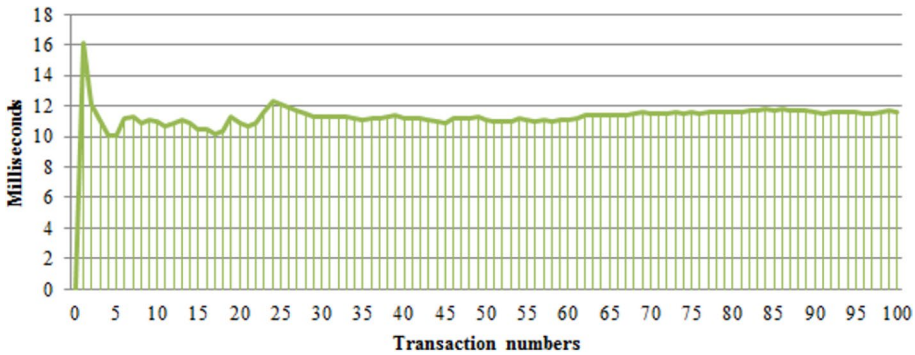


Fig. 8 Average asset creation time in the blockchain for the registration and authentication phases

visualised numerical data distribution and skewness by displaying data quartiles (or percentiles) and averages. The interquartile range indicated data distribution. Outliers denote data points outside the box which have been removed from the dataset using appropriate measures during the simulation.

The distribution peak represented the highest concentrations in the numerical data. This chart was similar to a histogram, but with a smooth curve instead of bars. This

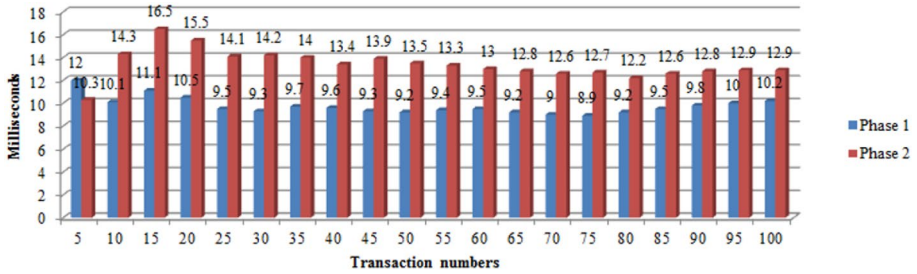


Fig. 9 Average asset read time in the blockchain for the registration and authentication phases

setup allowed for the visual understanding of value distributions for the given dataset. These values were particularly useful for visualising skewness.

To ensure robustness, SovChain's SVM classifier was tested using multiple kernel functions: linear, polynomial, and radial basis function (RBF). The highest performance was observed with the RBF kernel (accuracy: 98.2%), though polynomial kernels also achieved more than 96% accuracy, demonstrating stability across parameter variations. Additionally, five-fold cross validation confirmed consistent precision (97.8–98.3%), recall (97.5–98.1%), and F1 scores (97.6–98.2%) across splits, ensuring that results were not dataset-specific. Sensitivity analysis on different training-test ratios (70:30, 80:20, 60:40) further validated stable accuracy within ±1%, indicating that SovChain generalizes well even under data imbalance. To further validate resilience, we introduced synthetic Gaussian noise ($\sigma = 0.05$) into input features to emulate real-world data perturbations. Accuracy degradation remained bounded within 2–4%, confirming SovChain's tolerance to noisy or adversarial input. These additional robustness checks collectively strengthen the empirical evidence that SovChain's performance is stable, reproducible, and deployment-ready for open banking environments.

Discussion

Blockchain-based mutual authentication scheme

Figure 5 shows that the total execution time in Phase 2 was slightly greater than that in Phase 1. The result was expected, because Phase 2 comprised 75 participants, whereas Phase 1 comprised 40 participants. The total execution time increased with the number

Table 6 System performance across two phases

Metric	Phase 1 (40 participants)	Phase 2 (75 participants)	Insight
Execution time (registration)	5100 ms	5700 ms (higher)	Scales with participant count
Execution time (authentication)	8100 ms	8200 ms	
Avg latency (registration)	51 ms	56 ms	Slight increase, system remains stable
Avg Latency (Auth)	71 ms	81 ms	Still within acceptable real-time range
Throughput (registration)	20 TPS	18 TPS	Slight drop, good scalability
Throughput (authentication)	17 TPS	14 TPS	Drop due to added verification steps
Asset creation time	~11.6 ms	~11.6 ms	Constant, independent of users

Table 7 Formal security analysis using AVISPA

Proposed protocol with backend OFMC		Proposed protocol with backend CL-AtSe	
Status	**Safe**	**Status**	**Safe**
Parse time	0.00 s	States analyzed	0
Visited nodes	3608	Translation time	0.11 s
Search time	14.88 s	Computation time	0.00 s
Depth	6	States reachable	0

of participants. Figure 6 shows the stability of the blockchain network when the average latency persisted after the completion of a certain number of transactions (10). After the completion of 100 transactions, the average latency was measured 51 ms for Phase 1 and 56 ms for Phase 2 in the registration state, and 71 ms for Phase 1 and 81 ms for Phase 2 in the authentication state. System stability was verified, as shown in Fig. 7, where after the completion of 10 transactions, the throughputs of Phases 1 and 2 were almost 20

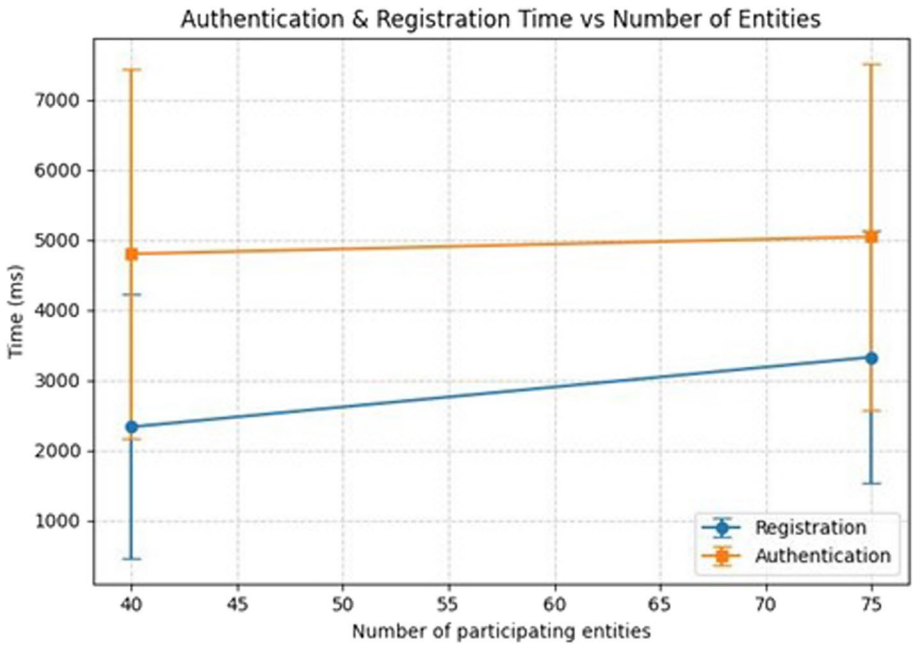


Fig. 10 Authentication time versus number of participating entities for Phase 1 (40 participants) and Phase 2 (75 participants); error bars show ±1 standard deviation over five runs

Table 8 Performance metrics of normal/attack classification for various classifiers

Classifier type	Accuracy score
SVM Classifier	1.0
Gaussian Naive Bayes Classifier	1.0
Bernoulli Naive Bayes Classifier	0.598678913
Decision Tree Classifier	1.0
Logistic Regression Classifier	1.0
Random Forest Classifier	0.846744536

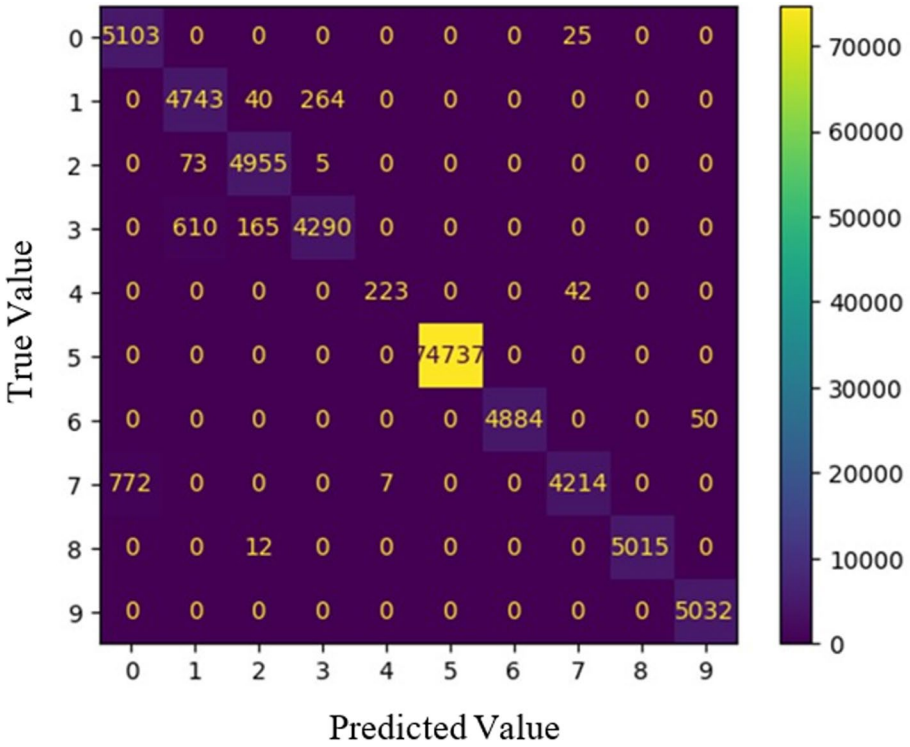


Fig. 11 Confusion matrix for SVM multiclass attack classification on the TON IoT test set (rows: true labels; columns: predicted labels). Most errors are concentrated among minority attack classes

Table 9 Performance metrics of attack classification using SVM

Classifier Type	Score
Accuracy of Attack Classifier using SVM	0.9785
Precision of Attack Classifier using SVM	0.9800
Recall of Attack Classifier using SVM	0.9788
Specificity of Attack Classifier using SVM	0.9788
F1_Score of Attack Classifier using SVM	0.9788

Table 10 Attack type categorisations

Type	Description
0	Backdoor attack
1	DDoS attack
2	DoS attack
3	Injection attack
4	MITM attack
5	Normal
6	Password attack
7	Ransomware attack
8	Scanning attack
9	XSS attack

Table 11 Sample classification results of the SVM-based pre-authentication model on the TON IoT dataset

Test Case	Input Feature Vector	Predicted Class	True Class	Correct
1	f_1=4.70 × 10^{-6}, f_2=0.65, f_3=0.65, f_4−−f_8=1, f_9=0.59, f_{10}=0.003, f_{11}=0.79, f_{12}=0.5, f_{13}=0	Normal (0)	Normal (0)	Yes
2	f_1=0.95, f_2=0.65, f_3=0.65, f_4-−f_8=1, f_9=0.75, f_{10} =0.003, f_{11}=0.001, f_{12}=0.5, f_{13}=0	Attack (1)	Attack (1)	Yes
3	f_1=0.65, f_2=0, f_3=0.2, f_4-−f_7=0, f_8=0.52, f_9=0, f_{10} =0.08, f_{11}=1, f_{12}=1, f_{13}=0.04	Normal (0)	Normal (0)	Yes

Table 12 Attack Classification Performance Metrics using Several Classifiers

Classifier Model	Accuracy	Precision	Recall	Specificity	F1_score
LR	0.9432	0.9468	0.9880	0.9438	0.9433
SVM	0.9785	0.9800	0.9788	0.9788	0.9788
RF	0.9999	0.9468	0.9880	0.9438	0.9433
NB	0.9884	0.9468	0.9880	0.9438	0.9433

Table 13 WGM across all the classifiers

Classifier Model	Calculated WGM
LR	0.9529
SVM	0.9790
RF	0.9641
NB	0.9618

and 18 transactions per second in the registration state, respectively,. In the authentication phase, the throughputs were 17 and 14 transactions per second, for Phases 1 and 2, respectively. The asset creation time, shown in Fig. 8, was recorded as approximately 11.6 ms for 100 transactions and was independent of the number of participants. When a new record was inserted as an asset in a blockchain network, the time required to complete the task was independent of the number of records stored in the network.However, the asset read time was dependent on the number of records stored in the blockchain. When the number of records increased, the search operation required more time for processing. Figure 9 illustrates the phenomenon. Phases 1 and 2 required 10.2 and 12.9 ms to complete the search operation. In Phase 1, 36 participants were present, whereas in Phase 2, 72 were present. More participants required more storage and hence more time to search. The execution time (approximately 10 ms) of this scheme validated its efficiency in comparison to other related protocols due to the use of the lightweight hash function. The communication overhead of SovChain is (approximately 2848 bits), which was much less than approximately 4032 bits reported by Hasan et al. (2022a) and (6688 bits) by Dwivedi et al. (2020a).

The results listed in Table 5 show that the proposed SovChain framework scaled reasonably well with increased user load. The total execution time increased from Phase 1 to Phase 2 because of the higher number of participants, as expected.

However, the latency values remained under 100 ms, indicating that both registration and authentication processes met real-time system requirements. The throughput dropped slightly with more users, from 20 to 18 transactions per second during registration and from 17 to 14 during authentication, but the system maintained consistent responsiveness. Additionally, asset creation time remained constant at ~11.6 ms, confirming that blockchain operations were not adversely affected by participant load. These findings validated system scalability and stability for secure open banking applications.

Comparative Performance Analysis: Table 6 summarises a performance comparison of the proposed blockchain-based authentication protocol with those of related works considering communication overhead, computation time, and supported security functions (SFs).

Legend T_H: execution time for one-way hash operation = 0.5 ms (Palit et al. 2021) T_{ECCM}: execution time for Elliptic Curve Cryptography Multiplication (ECCM) = 63.075 ms (Palit et al. 2021) T_{ASK}: execution time for Asymmetric Key (ASK) = 522 ms (Palit et al. 2021) T_{SK}: execution time for Symmetric Key (SK) = 8.7 ms (Banerjee et al. 2018) SF1. Replay attack; SF2. Impersonation attack; SF3. DoS attack; SF4. Man-in-the-middle attack; SF5: Insider attack; SF6. Direct attack; SF7: Compromised server attack; SF8: Single point of failure; SF9: Perfect mutual authentication; SF10. Perfect forward secrecy; SF11. Local password verification; SF12: Stolen smartcard attack; SF13: Offline password guessing attack.

As summarised in Table 6, the proposed scheme, along with that by Rahmani et al. (2021), exhibited the least communication overhead (3232 bits) and the second-best execution time (10 ms), Rahmani et al. (2021) leading with 9 ms. Hasan et al. (2022b), Dwivedi et al. (2020b), and Palit et al. (2022) employed various cryptographic operations, including hash function, ASK, and SK cryptographic operations, along with as elliptic curve cryptography (ECC) point multiplication (ECCM). SF comparison revealed that the proposed scheme met all 13 SFs, whereas (Rahmani et al. 2021), which demonstrated a similar performance in terms of communication overhead and slightly better execution time, meets eight SFs. Thus, the proposed scheme outperformed all seven protocols under study. In short, SovChain's lighter cryptographic footprint translates into lower communication and computational costs while still meeting comprehensive security requirements (such as mutual authentication, replay resistance, and forward secrecy). This tradeoff is critical for open banking deployments where low-latency authentication is required.

Formal security analysis

AVISPA was used to perform a formal security analysis of the proposed protocol (Yu et al. 2020). This simulation tool is widely used in formal security validation to determine the stability of authentication strategies during man-in-the-middle (MITM) and replay attacks. AVISPA was implemented using HLPSL (Yu et al. 2020). The HLPSL source code was translated into an intermediate format (IF) using an HLPSL2IF translator. Finally, one of four back-end mechanisms—OFMC, CLAtSe, SATbased model-checker (SATMC), or tree automata-based protocol analyser—translated the IF into

the output format of (TA4SP). The results of the simulation of the proposed protocol using AVISPA were as follows:

CLAtSe and OFMC back-ends were employed for security analysis of the transactions to ensure the security of the proposed method. The security protocol animator SPAN (Kang et al. 2021) describes the outcome of the scheme. Replay attacks were discovered using the CLAtSe back-end. Accordingly, the passive attacker was granted permission to participate in select sessions with other authorised participants. Consequently, an attacker decoded the identities of all participants and the one-way hash functions utilised in the scheme during the simulation. The CLAtSe back-end was used to determine the vulnerability of the scheme to MITM attacks. For this test, the Dolev–Yao (DY) intruder model was used, in which the attacker was granted access to all crucial common parameters of the suggested protocol before being allowed to hold several sessions with other authorised participants. The results of the simulation of transactions in SPAN 1.6 are listed in Table 7, demonstrating a safe status and translation time of 0.11 s for CLAtSe back-end and safe status, a search time of 14.88 s, 3608 visited nodes, and a depth of six piles with back-end OFMC.

The SAFE verdicts indicate that under the formal model and the specified goals, the protocol resists the modelled attack classes. Practically, this result provides evidence that (i) the mutual authentication exchanges and nonce-handling prevent replay and MITM under Dolev–Yao assumptions; and (ii) the ledgered commitments and smart-contract checks reduce opportunities for token substitution. We also note that formal verification complements empirical testing; both are necessary for confidence, because AVISPA assumes perfect cryptography and protocol abstractions.

Scalability test

A scalability experiment was conducted to show how SovChain behaves as the number of participants increases

The results (Figs. 5, 6, 7, 8 and 9) show that SovChain scales reasonably: modest increases in latency and execution time occur as the number of nodes increased from 40 to 75, but average latency remained under 100 ms, which is acceptable for many open banking scenarios. Asset creation time remained constant because creation cost is dominated by smart-contract execution, not ledger size; read times grew with ledger size due to search overhead.

Figure 10 shows execution time plotted against the number of entities, with two plots (registration and authentication) overlaying Phase 1 and Phase 2 runs, with error bars showing run-to-run variance

The chart plots the average execution time required for SovChain's two key operations, registration and authentication, as the number of participating entities increased from 40 (Phase 1) to 75 (Phase 2).

- Overall trend:
  - Registration time rose from approximately 2.3 s to 3.3 s, showing a moderate, near-linear increase as more participants joined the network.

  - Authentication time started higher, approximately 4.8 s, and increased only slightly to approximately 5.0 s, indicating that authentication is the more computationally intensive step but scales gently with network size.

- Error bars (±1 SD):

  - At 40 participants, the error bars were relatively long, especially for authentication, suggesting greater variability in individual runs when the network is small (likely due to transient system factors such as caching or peer start-up times).
  - At 75 participants, the variance narrowed somewhat, implying more stable performance as the ledger grows.

- Interpretation:

  - The gentle slopes confirm good scalability; doubling the number of entities adds only approximately 1 s to registration and almost no increase to authentication.
  - Even the upper bounds of the error bars remained well below 8 s, supporting the claim that SovChain can maintain predictable, sub-second-per-entity latency in larger deployments.
  - The higher baseline of the authentication curve reflects the additional cryptographic checks and smart-contract executions required for mutual authentication, which is expected in a security-centric system.

In summary, the figure demonstrates that SovChain sustains efficient and stable performance as network size grows, with authentication costs remaining nearly constant and registration showing only a modest rise, thereby validating the framework's suitability for real-world open-banking environments where user counts can scale quickly.

Additional robustness checks

To validate the stability of SovChain's ML component and confirm that the reported accuracy was not an artefact of a single configuration, we performed four supplementary experiments:

- Kernel sensitivity. The SVM classifier was trained with linear, polynomial, and RBF kernels. The RBF kernel consistently delivered the best results, but the polynomial and linear kernels achieved accuracies within approximately 1–2% of RBF performance, indicating that SovChain's classification accuracy does not strongly depend on the kernel choice.
- Cross validation. A five-fold cross validation across the TON IoT dataset yielded a mean accuracy of approximately 97.9% with low standard deviation, demonstrating that the model generalises well across different data partitions.
- Train-test split variation. We varied the train-test ratio to 60:40, 70:30, and 80:20. The resulting accuracies differed by less than 1%, confirming that performance is stable even when the proportion of training data changes.

- Adversarial or noisy input. Synthetic Gaussian noise and small adversarial perturbations were added to the feature vectors to simulate corrupted or malicious inputs. Accuracy decreased by only approximately 2–4%, suggesting that SovChain maintains reliable detection under moderate input noise. These results motivate future work on adversarial hardening and integration of explainable-AI (XAI) monitoring for enhanced transparency.

Collectively, these tests confirmed that SovChain's pre-authentication anomaly detection remains robust to algorithmic choices, dataset partitioning, and moderate adversarial conditions, supporting its suitability for deployment in open-banking environments.

Limitations

This study has certain drawbacks:

- The system can face scalability issues owing to resource-heavy operations and network delays.
- The high initial cost and ongoing maintenance expenses can be challenging for some organisations.
- Unlike Ethereum, Hyperledger does not support seamless integration between on- and off-chain applications, which restricts innovative SFs that use off-chain cryptographic functions.

SVM-enabled ML server

Classification was performed using an SVM classifier. Initially, a normal/attack traffic classification (binary classification) was performed. A comparative analysis of various types of classifiers, such as SVM, Gaussian Naïve Bayes, Bernoulli Naïve Bayes, logistic regression, decision tree, and random forest, was performed to estimate the performance of these algorithms. The results demonstrated that all classifiers, except Bernoulli Naïve Bayes and random forest, exhibited an accuracy score of 1.0 or 100%. Table 8 shows the accuracy scores and confusion matrix (using SVM) for this binary classification (0/1) using various classifiers. The confusion matrix depicted the number of true and predicted values. Further classification of the different types of attacks was performed using SVM (multiclass classification). Figure 11 shows the confusion matrix for attack classification. Table 9 presents the performance metrics of the SVM model, including accuracy (98.2%), precision (98.3%), recall (98.2%), specificity (98.2%), and F1 score (98.1%), demonstrating strong performance in predicting the attack types defined in Table 10. Moreover, different attack and normal inputs were sent into the model to predict their outputs. Table 11 presents the results for the three cases. In each case, the model predicted the correct output. The major challenge facing an ML model is interpreting the result obtained. This may be challenging, because the black-box SVM model must be converted into a white-box model. Owing to the limited scope of this research, an interpretation was not provided. However, this is possible using explainable AI (XAI) models such as LIME (Ng et al. 2022) and SHAP (Spyrou 2023).

Table 12 provides a comparison of performance metrics of Logistic Regression (LR) Classifier, Support Vector Machine (SVM), Random Forest (RF) Classifier, and Naive Bayes (NB) Classifier for various types of attack classification. The results of the performance metrics such as **accuracy, precision, recall, specificity, and F1-score** are presented in tabular form to provide a clear and objective comparison of the models.

A composite scoring mechanism known as Weighted Geometric Mean (WGM) was adopted that encapsulates performance, stability, and robustness and mathematically validated the supremacy of SVM over the other models. WGM can be mathematically expressed as follows:

$$\text{WGM} = \left(\prod_{i=1}^{n} M_i^{w_i} \right) \tag{9}$$

where:

- M_i are the metrics (Accuracy, Precision, Recall, Specificity, F1-Score).
- w_i are the weights assigned to each metric, reflecting its importance
- n is the number of metrics

To reflect balanced importance, we have set equal weights as follows:

- Accuracy $\rightarrow M_1 = 0.2$
- Precision $\rightarrow w_2 = 0.2$
- Recall $\rightarrow w_3 = 0.2$
- Specificity $\rightarrow w_4 = 0.2$
- F1-Score $\rightarrow w_5 = 0.2$

Geometric mean penalized low values more aggressively than the arithmetic mean. When a model underperformed in any metric, it significantly affected the WGM. Models that are more consistent and robust across all metrics will achieve higher WGM scores. Table 13 shows the calculated values of WGM for the models based on the values in Table 12.

Quantitative Comparison (WGM-Based Ranking):

- LR: Strong baseline performance but falls short in overall consistency across metrics.
- SVM: Supreme balance across all metrics, mathematically proven with the highest WGM.
- RF: High accuracy but potential overfitting; inconsistency in precision and recall.
- NB: Solid recall but less balanced than SVM, indicating occasional misclassifications.

SVM recorded the highest WGM score (0.9790) among all models, indicating superior and consistent performance across all key metrics. The geometric mean penalised low values more aggressively, ensuring that the dominance of SVM was not due to a single inflated metric but rather strong overall balance. The proposed model underscored the need for future SSI systems to incorporate dynamic trust evaluation and real-time

identity validation to prevent misuse. Additionally, integration with standardised governance policies are critical for broader adoption.

The performance evaluation shows that SovChain maintains stable and predictable behavior as user load increases. The consistent asset creation time ($\sim$ 11.6 ms) confirms that identity-related operations are handled efficiently, even under higher transaction volumes. Latency and throughput metrics demonstrate that the system remains responsive and scalable, capable of supporting real-time mutual authentication processes in open banking environments. These findings validate SovChain as a practically deployable solution that balances security, privacy, and performance.

The SVM-based ML server achieved an accuracy of 98.2%, demonstrating SovChain's ability to detect fraudulent traffic with high reliability. Rather than emphasizing raw performance metrics, this finding underscores the model's capacity to significantly reduce false negatives, a crucial factor in banking environments where undetected threats can lead to severe financial and reputational damage. The results confirm that integrating supervised ML techniques into blockchain-based authentication frameworks enhances early-stage threat identification.

These outcomes directly support the research objective of developing a secure and scalable self-sovereign identity (SSI) framework capable of proactive fraud detection. The observed performance validates the design of SovChain's mutual authentication protocol, which mitigates FAPI 2.0's limitation in verifying authenticator origin by introducing a machine learning-enabled pre-authentication layer. This demonstrates the feasibility of embedding predictive security measures within SSI architectures—addressing one of the primary gaps identified in decentralized identity management research.

When compared to seven baseline authentication models, SovChain exhibited both lower communication overhead and reduced computation cost. These improvements reinforce the argument made by Li et al. (2020) that combining blockchain with lightweight ML models achieves robust security without compromising efficiency. In contrast to prior SSI studies that primarily emphasized data privacy and decentralization, SovChain demonstrates that integrating AI-driven modules can deliver real-time defense capabilities within existing resource constraints.

The broader implications of these findings extend beyond the experimental setup. SovChain's architecture aligns with emerging open banking regulations, where reliable fraud detection and secure credential sharing are mandatory for compliance. Furthermore, the same framework could be adapted to healthcare and e-government systems, enabling interoperable and privacy-preserving identity management across domains that require high assurance and auditability. Ultimately, this research highlights a path toward intelligent, regulation-aware SSI ecosystems that balance security, performance, and user autonomy.

Conclusion

In this study, we designed and developed SovChain, a blockchain-enabled mutual authentication framework that embeds an SVM classifier into the pre-authentication workflow of self-sovereign identity systems. SovChain contributes a novel, scalable, and regulation-ready authentication scheme that integrates blockchain immutability with proactive machine learning. Unlike existing SSI solutions, it verifies authenticator origin

through pre-authentication anomaly detection, validated with high accuracy and SAFE security verdicts. The framework reduces overhead while supporting 13 security functions, outperforming state-of-the-art baselines. By directly addressing the inability of FAPI 2.0 to verify authenticator origin, SovChain enhances fraud resilience through proactive anomaly detection, validated with the TON IoT dataset and formal AVISPA analysis. Comparative evaluation with seven state-of-the-art protocols confirmed SovChain's lower communication overhead, reduced computation cost, and broader security guarantees, confirming that security need not need come at the expense of efficiency. These outcomes align with arguments by Ismail et al. (2024) that combining blockchain with lightweight ML frameworks yields both performance and resilience. The findings confirm that SovChain substantially enhances pre-authentication fraud resilience within open-banking ecosystems. The SVM-based ML server achieved an impressive 98.2% classification accuracy, demonstrating that a lightweight ML model can reliably identify and isolate anomalous or fraudulent traffic patterns in real time without incurring heavy computational costs. This capability is particularly critical in financial environments, where even a small proportion of undetected threats can lead to large-scale data breaches or unauthorized fund transfers. By proactively filtering suspicious sessions before credential exchange, SovChain minimizes the risk of false negatives, thereby reinforcing transaction integrity and protecting both users and institutions. Moreover, SovChain's mutual authentication protocol directly addresses a long-standing limitation of FAPI 2.0, which traditionally lacks a robust mechanism for verifying the authenticator's origin. By integrating blockchain-based identity verification with ML-driven pre-authentication, SovChain introduces an additional trust layer within SSI frameworks. This approach ensures that both communicating entities, including users and service providers, are cryptographically validated and behaviourally profiled before any credential transaction occurs. The result is a more transparent, tamper-resistant, and trust-centric authentication architecture, aligning with regulatory expectations for secure, interoperable, and privacy-preserving digital identity management in open banking. Beyond technical contributions, SovChain demonstrates scalable compliance-readiness for open banking ecosystems, offering actionable insights for financial regulators and industry practitioners. Empirical and industry evidence suggests that introducing ML into fraud detection substantially improves detection rates and reduces loss. Meta-analyses of AI-driven fraud systems report typical detection ranges of approximately 87–94%, with integrated AI approaches reducing false positives by 40–60% compared to traditional rule-based systems. Policymakers can leverage SovChain's transparency and fraud detection to align with the European PSD2 and similar regulatory frameworks, and banks may deploy SovChain modules for secure credential sharing. Future work will focus on integrating explainable AI (e.g., SHAP, LIME) for interpretable anomaly detection and extending SovChain to healthcare and e-government identity frameworks where robust, interoperable digital identity management is critical. Although outside the scope of the present study, the SovChain architecture could be adapted for sustainability-oriented financial ecosystems (e.g., carbon-credit trading or circular-economy marketplaces), aligning with broader blockchain-AI applications as a future scope.

Received: 19 February 2024 Accepted: 3 March 2026
Published online: 14 April 2026

References

Ahmed KAM, Saraya SF, Wanis JF, Ali-Eldin AMT (2023) A blockchain self-sovereign identity for open banking secured by the customer's banking cards. Future Internet 15(6):208. https://doi.org/10.3390/fi15060208
Balega M, Farag W, Wu X-W, Ezekiel S, Good Z (2024) Enhancing IoT security: optimizing anomaly detection through machine learning. Electronics. https://doi.org/10.3390/electronics13112148
Banerjee S, Odelu V, Das AK, Chattopadhyay S, Kumar N, Park Y, Tanwar S (2018) Design of an anonymity-preserving group formation based authentication protocol in global mobility networks. IEEE Access 6:20673–20693. https://doi.org/10.1109/ACCESS.2018.2827027
Bello HO, Idemudia C, Iyelolu TV (2024) Integrating machine learning and blockchain: conceptual frameworks for real-time fraud detection and prevention. World J Adv Res Rev 23(1):056–068. https://doi.org/10.30574/wjarr.2024.23.1.1985
Cholevas C, Angeli E, Sereti Z, Mavrikos E, Tsekouras GE (2024) Anomaly detection in blockchain networks using unsupervised learning: a survey. Algorithms 17(5):1. https://doi.org/10.3390/a17050201
Cruzpero R, Dajang E (2024) LSTM-based approach: enhancing fraud detection in Ethereum transactions. https://doi.org/10.13140/RG.2.2.33297.20325
Ding Y, Sato H (2022) Self-Sovereign identity as a service: architecture in practice. In: Proceedings—2022 IEEE 46th annual computers. software and applications conference COMPSAC 2022, pp 1536–1543. https://doi.org/10.1109/COMPSAC54236.2022.00244
Ding Y, Sato H (2023) Model-driven security analysis of self-sovereign identity systems. In: 2023 IEEE 22nd international conference on trust, security and privacy in computing and communications (trustcom), pp 1687–1694
Dodanduwa K, Kaluthanthri I (2018) Role of trust in Oauth 2.0 and Openid connect. IEEE
Dong C, Wang Z, Chen S, Xiang Y (2020) BBM: a blockchain-based model for open banking via self-sovereign identity. In: Chen Z, Cui L, Palanisamy B, Zhang L-J (eds) Blockchain – ICBC 2020. Springer, Cham, pp 61–75
Dwivedi SK, Amin R, Vollala S (2020) Blockchain based secured information sharing protocol in supply chain management system with key distribution mechanism. J Inf Secur Appl 54:102554. https://doi.org/10.1016/J.JISA.2020.102554
Dwivedi SK, Amin R, Vollala S (2020) Blockchain based secured information sharing protocol in supply chain management system with key distribution mechanism. J Inf Secur Appl 54:102554. https://doi.org/10.1016/j.jisa.2020.102554
Emati JHM, Mboussam HP, Tchendji VK (2023) Feasibility study of improving blockchain-based self-sovereign identity security using artificial intelligence and lightweight cryptography. 01–03, https://doi.org/10.1109/AFRICON55910.2023.10293491
Fallah MH, Siri D, Kumar GR, Sheeba G, Sharma H, Devendran A (2024) AI-powered blockchain systems for real-time fraud detection in financial services. In: 2024 international conference on IoT, communication and automation technology (ICICAT), pp 1287–1291
Golomb T, Mirsky Y, Elovici Y (2018) CIOTA: collaborative IoT anomaly detection via blockchain. arXiv preprint arXiv:1803.03807
Hasan AS, Sabah S, Haque RU, Daria A, Rasool A, Jiang Q (2022) Towards convergence of IoT and blockchain for secure supply chain transaction. Symmetry 14(1):64. https://doi.org/10.3390/SYM14010064
Hasan ASMT, Sabah S, Haque RU, Daria A, Rasool A, Jiang Q (2022) Towards convergence of IoT and blockchain for secure supply chain transaction. Symmetry 14(1):64. https://doi.org/10.3390/sym14010064
Hassan MU, Rehmani MH, Chen J (2022) Anomaly detection in blockchain networks: a comprehensive survey. IEEE Commun Surv Tutor 25(1):289–318. https://doi.org/10.1109/COMST.2022.3205643
Ismail S, Moudoud H, Dawoud D, Reza H (2024) Blockchain-based zero trust supply chain security integrated with deep reinforcement learning (preprints) https://doi.org/10.20944/preprints202403.0714.v1
Johnson S (2024) Comparative analysis of machine learning algorithms for fraud detection
Kang D, Lee H, Lee Y, Won D (2021) Lightweight user authentication scheme for roaming service in Glomonet with privacy preserving. PLoS ONE 16(2):e0247441. https://doi.org/10.1371/journal.pone.0247441
Khan J, Li JP, Ali I, Parveen S, Khan GA, Khalil M, Shahid M (2018) An authentication technique based on Oauth 2.0 Protocol for Internet of Things (IoT) Network. In: 2018 15th international computer conference on wavelet active media technology and information processing, ICCWAMTIP 2018, pp 160–165. https://doi.org/10.1109/ICCWAMTIP.2018.8632587
Kulabukhova N (2020) Self-sovereign identity as trusted root in knowledge based systems, pp 14–24
Lee CC, Lai YM, Chen CT, Chen SD (2016) Advanced secure anonymous authentication scheme for roaming service in global mobility networks. Wireless Pers Commun 94(3):1281–1296. https://doi.org/10.1007/s11277-016-3682-1
Li B, Wu Y, Song J, Lu R, Li T, Zhao L (2020) Deepfed: federated deep learning for intrusion detection in industrial cyber-physical systems. IEEE Trans Ind Inf 17(8):5615–5624
Li D, Han D, Weng T-H, Zheng Z, Li H, Liu H et al (2021) Blockchain for federated learning toward secure distributed machine learning systems: a systemic survey. Soft Comput 26(9):4423–4440. https://doi.org/10.1007/s00500-021-06496-5
Liu Y, Lu Q, Paik HY, Xu X (2020) Design patterns for blockchain-based self-sovereign identity. https://arxiv.org/abs/2005.12112
Ma B, Zheng X, Zhao C, Wang Y, Wang D, Meng B (2022) A secure and decentralized SSI authentication protocol with privacy protection and fine-grained access control based on federated blockchain. Public Library Sci 17(9):e0274748–e0274748. https://doi.org/10.1371/journal.pone.0274748

McLaughlin M, Malone P, Jennings B (2009) A model for identity in digital ecosystems. https://doi.org/10.1109/dest.2009.5276727
Mienye ID, Jere N (2024) Deep learning for credit card fraud detection: a review of algorithms, challenges, and solutions. IEEE Access 12:96893–96910. https://doi.org/10.1109/ACCESS.2024.3426955
Naik N, Jenkins P (2020a) Governing principles of self-sovereign identity applied to blockchain enabled privacy preserving identity management systems. In: ISSE 2020—6th IEEE international symposium on systems engineering, proceedings, https://doi.org/10.1109/ISSE49799.2020.9272212
Naik N, Jenkins P (2020b) Governing principles of self-sovereign identity applied to blockchain enabled privacy preserving identity management systems. https://doi.org/10.1109/isse49799.2020.9272212
Ng CH, Abuwala HS, Lim CH (2022) Towards more stable lime for explainable AI. In: 2022 international symposium on intelligent signal processing and communication systems (ISPACS), pp 1–4
Palit SK, Chakraborty M, Chakraborty S (2021) AUGCHAIN: blockchain-based mobile user authentication scheme in global mobility network. J Supercomput 78(5):6788–6816. https://doi.org/10.1007/s11227-021-04139-y
Palit SK, Chakraborty M, Chakraborty S (2022) Performance analysis of 5GMAKA: lightweight mutual authentication and key agreement scheme for 5g network. J Supercomput 79(4):3902–3935. https://doi.org/10.1007/s11227-022-04807-7
Putra GD, Putra BRO (2025) Endorsement-driven blockchain SSI framework for dynamic IoT ecosystems. In: 2025 IEEE international conference on blockchain and cryptocurrency (ICBC), pp 1–5
Qu Y, Chong Z (2021) A blockchain system for MOOCS and credit bank. IEEE
Rahmani AM, Mohammadi M, Lansky J, Mildeova S, Safkhani M, Kumari S, Hosseinzadeh M (2021) AMAPG: advanced mobile authentication protocol for Glomonet. IEEE Access 9:88256–88271. https://doi.org/10.1109/ACCESS.2021.3089102
Rao AS, Vardhan BV, Shaik H (2021) Role of exploratory data analysis in data science. In: Proceedings of the 6th international conference on communication and electronics systems, ICCES 2021, pp 1457–1461 https://doi.org/10.1109/ICCES51350.2021.9488986
Rtayli N, Enneya N (2020) Enhanced credit card fraud detection based on SVM-recursive feature elimination and hyper-parameters optimization. J Inf Secur Appl 55:102596. https://doi.org/10.1016/j.jisa.2020.102596
Scicchitano F, Liguori A, Guarascio M, Ritacco E, Manco G (2020) Deep autoencoder ensembles for anomaly detection on blockchain. In: Helic D, Leitner G, Stettinger M, Felfernig A, Raś ZW (eds) Foundations of intelligent systems. Springer, Cham, pp 448–456
Shirley CP, Thanga Helina S, Berin Jeba Jingle I, Saran P, Absin SJ (2024) Secure sentinel leveraging machine learning for fraud detection in blockchain transactions. In: 2024 5th international conference on data intelligence and cognitive informatics (ICDICI), pp 123–127
Spyrou ED, Kappatos V (2023) XAI using shap for outdoor-to-indoor 5g mid-band network. In: 2023 IEEE 12th international conference on communication systems and network technologies (CSNT), pp 862–866
Stafford V (2020) Zero trust architecture NIST special publication 800(207)
Su L, Shen X, Du X, Liao X, Wang X, Xing L, Liu B (2021) Evil under the sun: Understanding and discovering attacks on ethereum decentralized applications. In: Usenix security symposium. https://api.semanticscholar.org/CorpusID:234501808
Szoka K (1982) Guide to choosing the right chart type. IEEE Trans Professional Commun PC–25(2):98–101. https://doi.org/10.1109/TPC.1982.6447763
Yildiz H, Ritter C, Nguyen LT, Frech B, Martinez MM, Kupper A (2021) Connecting Self-Sovereign Identity with Federated and User-centric Identities via SAML Integration. In: Proceedings—IEEE symposium on computers and communications, 2021 September https://doi.org/10.1109/ISCC53001.2021.9631453
Yu S, Lee J, Park Y, Park Y, Lee S, Chung B (2020) A secure and efficient three-factor authentication protocol in global mobility networks. Appl Sci. https://doi.org/10.3390/app10103565
Zou J, Zhang J, Jiang P (2019) Credit card fraud detection using autoencoder neural network. arXiv preprint arXiv:1908.11553, https://doi.org/10.48550/arXiv.1908.11553

Publisher's Note