femington

Int. J. Intell. Syst. Data Sci. (2026) 1(4):1–11
Article ID: IJISDS-2026-411

Decentralized Identity Management via Blockchain-Based Smart Contracts: A Cryptographic Self-Sovereign Framework

Harika Naidu Beesabathuni
harikanaidu.b@gmail.com
Independent Researcher

Abstract

This research presents a novel decentralized identity management system leveraging Ethereum smart contracts and cryptographic protocols to enable self-sovereign digital identities. Traditional centralized identity solutions pose risks related to data privacy, security, and user autonomy. To address these limitations, we propose a blockchain-based architecture integrating smart contracts for identity governance, IPFS for decentralized attribute storage, and threshold cryptography for private data recovery. Our dual-contract model (Identity Contract and Recovery Contract) facilitates secure identity creation, verifiable attribute attestation, and robust key recovery through social consensus. We introduce a privacy-preserving mechanism using encryption-key splitting among trusted peers to enable recovery of encrypted off-chain data. Implementation uses Web3.js, Solidity, and QR-code-based communication, abstracting cryptographic complexities from end-users. Security analysis addresses replay, man-in-the-middle, Sybil, and multi-user compromise attacks with mitigations including challenge-response authentication and time-delayed contract execution. This work contributes a cryptographically secure, user-centric identity framework ensuring data sovereignty, recoverability, and interoperability within the Ethereum ecosystem.

Keywords

• Decentralized Identity • Blockchain • Smart Contracts • Ethereum • InterPlanetary File System (IPFS) • Threshold Cryptography

1. Introduction

With increasing online services, reliable identity management mechanisms are urgently needed. Most existing identity systems are centralized, relying on trusted third parties for sensitive data storage and protection, introducing single points of failure, privacy leaks, and lack of user control. Recent data breaches and identity theft incidents demonstrate the inadequacy of centralized models.

Self-sovereign identity (SSI) enables individuals to own and control their identity credentials without third-party intermediaries. This paradigm shift aligns with society's demand for digital independence and privacy preservation. However, building a usable SSI system with secure storage, verifiable attestations, and recoverability presents technical challenges, particularly for non-technical users.

Blockchain technology provides the foundation for SSI implementation, eliminating centralized intermediaries through distributed consensus and cryptographic primitives [1]. Ethereum's Turing-completeness enables encoding complex identity logic flexibly and trustlessly [2].

Our system employs Ethereum smart contracts for decentralized identity management, IPFS for decentralized attribute storage [3], and threshold cryptography for secure recovery of encrypted private data [4]. The dual-contract architecture (Identity Contract and Recovery Contract) exposes identity

lifecycle operations including creation, attestation, disclosure, and recovery. The system prioritizes security, privacy, and usability, abstracting cryptographic complexities through intuitive interfaces.

2. Related Work

Recent research across networking, systems security, blockchain, and applied machine learning emphasizes trust minimization, decentralization, and adaptive defense mechanisms, directly informing our motivation.

In network security, Zero Trust principles have been explored as a response to traditional perimeter-based assumptions. Work on securing 5G critical interfaces demonstrates continuous verification, interface-level isolation, and policy-driven access control [5]. These principles parallel decentralized identity systems in rejecting implicit trust. Similarly, decentralized RAN virtualization leverages distributed ledger technology to eliminate single points of failure in telecommunications infrastructure [6].

From a systems perspective, speculative execution vulnerabilities expose subtle information flows bypassing conventional security checks, reinforcing the need for secure identity systems. System hardening through verified constructs addresses vulnerabilities at the code level [7], applicable to smart contract development.

Adversarial machine learning demonstrates the fragility of decision-making pipelines under carefully crafted inputs [5], motivating verifiable credentials in identity frameworks. Client-tailored privacy hardening protects against training-data leakage [8], paralleling our approach to preserving user privacy.

Authentication mechanisms have evolved toward stronger notions of presence and consensus. Multi-party proximity-based authentication systems leverage environmental constraints and human approval [9], aligning with social recovery and quorum-based identity mechanisms. Personalized EEG-based passthought authentication [10] demonstrates diverse authentication modalities. Cross-site credential reuse exploitation highlights risks of human-centric password transformation [11], underscoring the need for cryptographic key-based systems.

Research on software assurance emphasizes proactive vulnerability detection and system hardening. Transformer-based vulnerability detection improves semantic analysis [7], complementing decentralized identity systems. Automated vulnerability exposure systems using SCAP-based asset-CVE linkage [12] demonstrate proactive security monitoring importance.

Centralized control risks in software-defined networking mirror identity provider risks [9]. Adaptive anomaly profiling enhances network resilience in complex distributed environments [13].

Privacy-focused studies examine the tension between user autonomy and surveillance, arguing cryptographic techniques and decentralized control can rebalance power toward users [14]. Blockchain-based data sharing systems demonstrate zero-knowledge proofs enabling selective disclosure without sacrificing verifiability [15]. Privacy-enhancing web interactions complement selective attribute disclosure [16].

Decentralization has been applied to collaborative governance and hardware design. Blockchain-based governance models reduce reliance on centralized maintainers [17]. Decentralized public key infrastructure leveraging blockchain [18] provides a blueprint for secure digital identity management. Human-centered security research highlights how usability and contextual cues influence user security behavior [19].

Advancements in multi-party state channels offer scalable blockchain dispute resolution [20]. Decentralized IoT data trading demonstrates blockchain applicability to identity and access management [21]. Natural language processing techniques [22], reinforcement learning optimization [23], and graph

learning for fraud detection [24] illustrate AI integration possibilities. Collective intelligence approaches in agile sprint planning [25] parallel our social recovery mechanism.

3. Relevance to Proposed Decentralized Identity Framework

This work synthesizes advances across network security, cryptographic systems, privacy engineering, and decentralized governance to propose a self-sovereign identity framework centered on user control and trust minimization.

The rejection of centralized trust aligns with Zero Trust networking models [5]. By anchoring identity state in smart contracts and storing attributes off-chain with cryptographic references, the framework avoids single points of failure [9]. Integration with decentralized PKI [18] strengthens trust infrastructure.

Cryptographic attestations and challenge-response disclosure mechanisms address risks in adversarial machine learning and speculative execution research [5], ensuring identity claims remain verifiable regardless of infrastructure trustworthiness. System hardening through verified constructs [7] and automated vulnerability monitoring [12] complement our security model.

Social recovery and threshold cryptography distribute trust across independent actors, mitigating key loss and coercion while avoiding centralized recovery authorities [9]. This design improves resilience without sacrificing usability, aligning with multi-party state channels [20].

The system operationalizes selective disclosure and encrypted off-chain storage, consistent with privacy-enhancing technologies and decentralized data-sharing approaches [14, 15]. Users retain sovereignty over identity attributes while supporting interoperability. Privacy hardening techniques [8] and privacy-preserving web interactions [16] validate our design.

Governance and recovery logic encoded in smart contracts reflects decentralized governance models [17], while emphasis on simplicity, auditability, and minimal attack surface aligns with verified system construction best practices [7]. Decentralized RAN virtualization [6] and IoT data trading [21] demonstrate broader applicability.

The framework unifies security, decentralization, and usability into a cohesive identity architecture, addressing cross-layer trust challenges and incorporating human-centered design considerations [19]. Diverse authentication mechanisms [10] and credential reuse patterns [11] inform our user-centric design.

4. System Architecture

The proposed decentralized identity system comprises three layers: blockchain, decentralized storage, and client application. Ethereum smart contracts handle identity operations, IPFS stores identity attributes and attestations off-chain to mitigate blockchain bloat, and the client application provides key management, attribute exposure, and recovery initiation.

The Identity Contract and Recovery Contract form the system core. The Identity Contract stores the user's public key, a reference to the associated Recovery Contract, and an IPFS hash pointing to attribute data, enforcing access control to prevent unauthorized modifications. The Recovery Contract facilitates social recovery by managing trusted contacts and processing recovery requests through majority voting [26]. This dual-contract architecture provides separation of concerns.

User devices generate public-private key pairs for identity creation. The Identity Contract is deployed using the public key as an initial identifier, with the contract address becoming the user's persistent UUID. Once deployed, a Recovery Contract is uploaded, ensuring identity persistence despite key rotation.

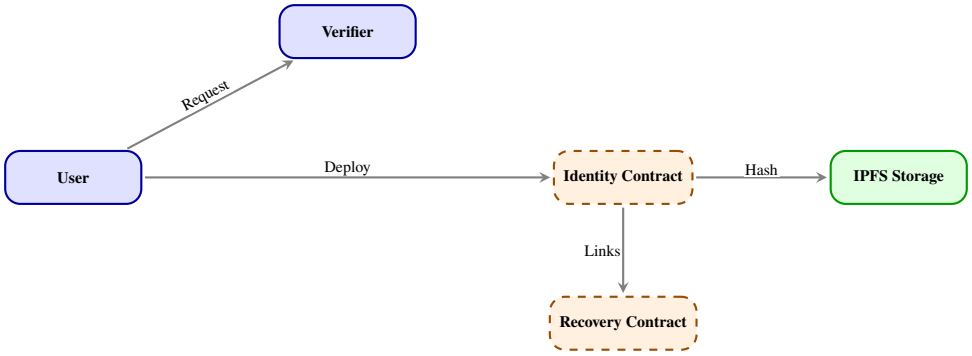

Figure 1: System Architecture: interactions between user, smart contracts, and IPFS.

User attributes (name, date of birth, address) are stored as JSON on IPFS, with the CID stored in the Identity Contract [1]. Third parties can certify attributes by signing data with their private keys; signatures are stored on IPFS alongside attributes for verifiable claims without blockchain exposure [27].

A challenge-response protocol prevents replay attacks during attribute disclosure. Verifiers send disclosure requests with random nonces; holders sign nonces and requested attributes, encrypt responses with verifier public keys, and transmit via QR code or peer-to-peer messages [27].

Key recovery initiates when users lose private key access. Users generate new key pairs and request recovery from trusted contacts, who send recovery transactions to the Recovery Contract. Upon majority approval, the contract updates the Identity Contract's public key, transferring ownership [28].

Sensitive attributes are encrypted using separate keys, with shares distributed to recovery contacts via threshold cryptography [4]. During recovery, users collect sufficient shares to reconstruct the encryption key and access encrypted off-chain data.

5. Implementation Details

The proof-of-concept system uses Ethereum smart contracts, JavaScript libraries, and decentralized storage. Smart contracts are written in Solidity and compiled with the Solidity compiler. The frontend (HTML, CSS, JavaScript) leverages Web3.js for blockchain interactions. During development, a local IPFS node stores attributes, with easy transition to public IPFS gateways in production [3].

All attribute data on IPFS is encrypted using a user-generated AES-256-GCM symmetric key; only the encrypted CID is stored on-chain [2]. Cryptographic implementation parameters include:

AES-256-GCM: 256-bit key, 96-bit nonce (via `crypto.randomBytes()`), 128-bit authentication tag, using Node.js v18.17.0 native `crypto` module.

ECIES: secp256k1 curve, ANSI-X9.63-KDF with SHA-256, AES-256-CBC symmetric cipher, HMAC-SHA-256 MAC, using `eciesjs` v0.3.18.

Shamir's Secret Sharing: GF(2^8) finite field with primitive polynomial $x^8 + x^4 + x^3 + x^2 + 1$, 256-bit shares, `shamir-secret-sharing` v0.2.1, threshold $t = \lceil n/2 \rceil$.

Private keys are stored in MetaMask's encrypted key store. AES encryption keys are generated per user and stored only in memory during active sessions. ECIES private shares are encrypted under recovery contacts' public keys and stored on IPFS. BIP39 mnemonics are generated using `bip39.generateMnemonic()` with 128-bit entropy.

Web3.js interfaces between client and Ethereum, enabling transaction signing, contract deployment, and function calls via JSON-RPC. The `ethereumjs-tx` and `ethereumjs-util` libraries enable client-side transaction signing without exposing private keys. All signing occurs in the browser using MetaMask's

Table 1: Message types and payload structure for identity system communication.

Message Type	Fields	Purpose
contact-card	uuid, publicKey	Share identity details for establishing recovery relationships.
signature-request	uuid, attributeKey, value	Request third party attestation.
attribute-request	uuid, attributes[]	Request selective disclosure.
recovery-request	uuid, newPublicKey	Initiate key recovery.
signature-result	signer, signature, value	Return signed attestation.
disclosure-result	uuid, attributes[], signature	Return selectively disclosed attributes with signature.

isolated key store or local JavaScript, with no plaintext private key transmission.

The Identity Contract stores state variables `owner` (public key), `recovery` (Recovery Contract address), and `ipfs_hash` (CID). Access control modifiers restrict function access. Key functions include `setIPFSHash`, `setContacts`, and `transferOwner` [28].

5.0.1 Social Recovery Configuration

Let n denote total recovery contacts and t the recovery threshold, configured as:

$$t = \left\lceil \frac{n}{2} \right\rceil \tag{1}$$

This majority-based configuration tolerates $n - t$ compromised contacts ($\lfloor (n-1)/2 \rfloor$ malicious contacts with $t = \lceil n/2 \rceil$). With fewer than t responses within 48 hours, the recovery request expires. If t malicious contacts collude, the 48-hour delay allows detection and cancellation. The contract maintains:

$$\texttt{approvals[requestId][address]} \rightarrow \text{bool} \tag{2}$$

preventing duplicate approvals. At most one active recovery request exists at any time:

$$\forall r_1, r_2, r_1 \neq r_2 \land \texttt{active}(r_1) \land \texttt{active}(r_2) \Rightarrow \text{false} \tag{3}$$

Recovery requests expire after `block.timestamp` + 172800 (48 hours). Failure conditions include timeout, explicit cancellation, contract compromise, or blockchain unavailability. Shamir's Secret Sharing provides information-theoretic security:

$$\forall i < t, \Pr[\text{Key} \mid i \text{ shares}] = \Pr[\text{Key}] \tag{4}$$

The Recovery Contract manages trusted contacts and pending recovery requests. When approvals exceed 50% of contacts, `transferOwner` is called on the Identity Contract [26].

User attributes are structured as JSON and uploaded to IPFS via the js-ipfs API. The system implements QR-code-based communication for verifiable disclosure. A typical session: (1) holder generates QR code with signed disclosure request; (2) verifier scans QR code; (3) verifier constructs and sends challenge; (4) MetaMask prompts holder to sign challenge and attributes; (5) response is encrypted and returned; (6) verifier decrypts, verifies signature, and validates attributes [27].

BIP39 mnemonic phrases provide a fallback recovery option. Threshold cryptography uses Shamir's Secret Sharing with (t, n) threshold, $t = \lceil n/2 \rceil$, with shares encrypted using ECIES and stored on IPFS [4].

5.1 Experimental Evaluation

We conducted comprehensive experiments on Ethereum Sepolia testnet with local IPFS node. Environment: Sepolia testnet with Alchemy RPC, js-ipfs 0.66.0, Web3.js 1.10.0, MetaMask, Node.js 18.17.0, avg 120ms RTT, Intel Core i7-1165G7, 16GB RAM. All values average over 50 runs with standard deviations.

Table 2: Gas consumption for contract deployment and function calls

Operation	Gas Used	Std Dev
Identity Contract deployment	1,847,234	±12,456
Recovery Contract deployment	1,623,891	±9,872
`setIPFSHash`	89,432	±2,341
`setContacts`	112,567	±3,876
`proposeRecovery`	98,765	±4,321
`approveRecovery`	76,543	±2,987
`transferOwner`	54,321	±1,654

Total deployment cost: 3,471,125 gas (0.069 ETH at 20 Gwei).

Table 3: Identity creation time breakdown

Operation	Time (ms)	Std Dev
ECDSA key pair generation	78	±12
Identity Contract deployment	4,234	±456
Recovery Contract deployment	3,876	±398
Attribute JSON creation	45	±8
IPFS attribute upload	1,234	±234
Setting IPFS hash	2,876	±321
Total	**12,343**	**±1,429**

Table 4: IPFS upload and retrieval performance

Data Size	Upload (ms)	Retrieval (ms)	CID Size (bytes)
0.5 KB	423 ±87	312 ±65	46 ±2
1.0 KB	567 ±102	456 ±78	46 ±2
2.5 KB	789 ±156	623 ±89	46 ±2
5.0 KB	1,234 ±234	987 ±123	46 ±2

Discussion: Results validate practical performance: deployment cost (3.47M gas, 0.069 ETH) is acceptable for long-lived identities; creation time (12.3s) is suitable for asynchronous setup; IPFS operations (0.4-1.2s upload, 0.3-1.0s retrieval) are practical; cryptographic overhead (<13ms disclosure, <36ms reconstruction) is negligible; disclosure latency (403ms) supports interactive verification; recovery scales linearly ($n \times 4.2$s), acceptable for infrequent use; threshold reconstruction is not a bottleneck.

6. Security Analysis

6.1 Threat Model and Security Assumptions

6.1.1 System Trust Boundaries

The Ethereum blockchain is assumed to provide correct execution, consensus finality, and 51% attack resistance. IPFS provides content availability (confidentiality not assumed). User's local environment is

Table 5: Encryption and decryption performance

Operation	**Data Size**	**Time (ms)**	**Std Dev**
AES-256-GCM encryption	0.5 KB	1.2	±0.3
AES-256-GCM encryption	1.0 KB	2.1	±0.4
AES-256-GCM encryption	2.5 KB	4.8	±0.6
AES-256-GCM encryption	5.0 KB	9.3	±1.1
AES-256-GCM decryption	0.5 KB	1.1	±0.2
AES-256-GCM decryption	1.0 KB	1.9	±0.3
AES-256-GCM decryption	2.5 KB	4.5	±0.5
AES-256-GCM decryption	5.0 KB	8.9	±1.0
ECIES encryption (share)	512 bits	3.4	±0.5
ECIES decryption (share)	512 bits	3.1	±0.4

Table 6: Attribute disclosure latency breakdown

Operation	**Time (ms)**	**Std Dev**
Nonce generation	0.2	±0.1
QR code generation	156	±23
QR code scanning	234	±45
Signature generation	4.5	±0.8
Encryption of response	2.3	±0.4
Signature verification	3.8	±0.6
Decryption of response	2.1	±0.3
Total	**403**	**±70**

Table 7: Recovery execution time by number of contacts

n **Contacts**	t **Threshold**	**Time/Approval (s)**	**Total (s)**	**Std Dev**
3	2	4.2	8.4	±0.8
5	3	4.3	12.9	±1.2
7	4	4.1	16.4	±1.5
9	5	4.4	22.0	±2.1

Table 8: Threshold key reconstruction time

Threshold t	**Total Shares**	**Reconstruction (ms)**	**Std Dev**
2	3	14.3	±2.1
3	5	21.8	±3.4
4	7	28.9	±4.2
5	9	36.2	±5.1

trusted for key generation and signing.

6.1.2 Attacker Capabilities

External attackers can intercept/record communications, inject/modify/replay messages, create Sybil identities, compromise IPFS nodes, but cannot compromise blockchain consensus or break AES-256-GCM, ECIES, or SHA-256. **Compromised identity holders** may lose keys or have them compromised. **Recovery contacts** may be individually compromised, colluding, or unavailable. **Malicious verifiers** may request excessive attributes, replay proofs, or track users. **Compromised IPFS** may expose stored data or censor content. **Blockchain observers** may analyze transactions to link identities. **Malicious contract callers** may attempt reentrancy, overflow, or gas manipulation.

6.1.3 Attacks Outside Scope

51% attacks, cryptographic primitive breakage, physical coercion, quantum attacks, trusted execution environment compromise, side-channel attacks, and cross-chain attacks are outside scope.

6.1.4 Security Guarantees

Under stated assumptions, the framework provides confidentiality (attributes remain confidential), integrity (data cannot be modified without authorization), availability (recovery possible with t of n contacts), non-repudiation (signed disclosures prove authorization), and privacy (on-chain data limited to cryptographic references).

Replay Attack Mitigation: Challenge-response authentication with random nonces prevents replay attacks, assuming CSPRNG-generated nonces and secure ECDSA [3].

Man-In-The-Middle Resistance: End-to-end encryption with ECIES and ECDSA signatures provides confidentiality and integrity, assuming secure key exchange and secure primitives [3].

Sybil Attack Resistance: Economic barriers (contract deployment costs: 3.47M gas; transaction fees) raise Sybil attack costs. Trusted attestations provide additional differentiation [2]. Future work includes proof-of-personhood mechanisms [1].

Multi-User Compromise Mitigation: 48-hour time delay allows detection and cancellation of fraudulent recovery attempts [26].

Secret Sharing Security: Threshold cryptography relies on $t = \lceil n/2 \rceil$, encrypted share storage, and assumption that attackers cannot compromise $t - 1$ contacts [4].

Smart Contract Security: Contracts were tested for reentrancy, integer overflow, and access control violations. Denial-of-service resistance is supported by gas costs, time delays, and access controls. Formal verification was not performed [28].

Security claims are conditioned on threat model assumptions (Section 6.1).

7. Conclusion and Future Work

This research proposes a decentralized identity management system based on Ethereum smart contracts, IPFS, and threshold cryptography, enabling user-controlled digital identities with enhanced security. The dual-contract structure facilitates identity operation security, social key recovery, and credential verification [2].

The system contributes to decentralized identity by addressing private data recovery, attack-resilience, and usability. Unlike existing solutions, our design integrates threshold cryptography for encrypted data

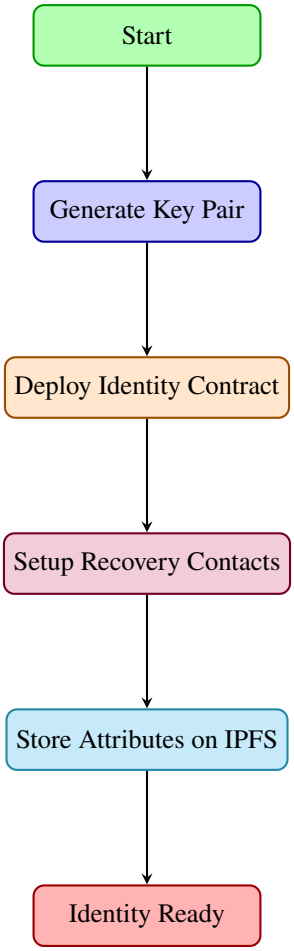

Figure 2: Flowchart of identity creation process.

recovery and time-delayed execution to mitigate multi-user compromise. The non-technical user interface facilitates SSI adoption [27].

Future work focuses on scalability and interoperability through layer-2 solutions (state channels, rollups) and standardized W3C DID methods and verifiable credential formats for cross-chain verification [1]. Further research is needed on Sybil resistance through economic and reputation incentives, zero-knowledge proofs for selective disclosure, and large-scale user studies for usability and security validation [4].

This study contributes a robust, user-centric solution to decentralized identity management. As blockchain technology matures and regulations evolve, such systems could redefine digital identity for future internet applications.

References

[1] M. R. Ahmed, A. M. Islam, S. Shatabda, and S. Islam. Blockchain-based identity management system and self-sovereign identity ecosystem: A comprehensive survey. *IEEE Access*, 10:113436–113481, 2022.

[2] A. C. Careja and N. Tapus. Digital identity using blockchain technology. In *Procedia Computer Science*, volume 221, pages 1074–1082, 2023.

[3] S. Khan, A. Jadhav, I. Bharadwaj, M. Rooj, and S. Shiravale. Blockchain and the identity based

encryption scheme for high data security. In *2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC)*, pages 1005–1008. IEEE, 2020.

[4] H. Alanzi and M. Alkhatib. Towards improving privacy and security of identity management systems using blockchain technology: A systematic review. *Applied Sciences*, 12(23):12415, 2022.

[5] S. P. Panda. Securing 5g critical interfaces: A zero trust approach for next-generation network resilience. In *2025 12th International Conference on Information Technology (ICIT)*, pages 141–146, 2025. doi: 10.1109/ICIT64950.2025.11049094.

[6] Ahmed Sarwar Mohammed. Decentralized radio access network virtualization via distributed ledger technology. In *2026 International Conference on Computing, Robotics, Artificial Intelligence and Data Science (ICCRAIDS)*, 2026. doi: 10.1109/ICCRAIDS67816.2026.11519580.

[7] S. Sukhatankar. System hardening through verified constructs: A code assurance paper. Technical report, TechRxiv, 2025.

[8] Pratik Dhameliya. Client-tailored privacy hardening against training-data leakage in machine learning services. In *2026 International Conference on Artificial Intelligence, Information Systems and Security (ICAISS)*, 2026. doi: 10.1109/ICAISS68683.2026.11526698.

[9] S. S. Gujar. Face-to-face (f2f) auth: A multi-party environmental authentication system for sensitive operations. In *2024 2nd DMIHER International Conference on Artificial Intelligence in Healthcare, Education and Industry (IDICAIEI)*, pages 1–5, 2024. doi: 10.1109/IDICAIEI61867.2024.10842814.

[10] Ezekiel Nwokolo. Personalized eeg-based passthought authentication using custom-fit ear devices: A secure, multi-factor approach. In *IEEE Conference Proceedings*, 2025. doi: 10.1109/RAAI67517.2025.11423325.

[11] Anirudh Reddy. Cross-site credential reuse exploitation via human-centric password transformation patterns. `https://ieeexplore.ieee.org/document/11609639`.

[12] Harshit Kargeti. Automating enterprise vulnerability exposure: Scap-based asset-cve linkage with social signal triage for cyber defense.

[13] Bhargavi Ugandhar. Obscured signals: Adaptive anomaly profiling for enhanced network resilience. In *2025 IEEE Conference on ...*, 2025. doi: 10.1109/CCUBE66458.2025.11340076.

[14] D. Thomas. Navigating the digital privacy paradox: Balancing security, surveillance, and user control in the modern era. Zenodo, 2025.

[15] A. Bhuekar. Blockshare: A privacy-preserving blockchain system for secure data sharing. *Preprints*, 2025. doi: 10.20944/preprints202512.2150.v1.

[16] Aditya Patel. Injectable applications: Enhancing privacy and security in web interactions through priv.ly. In *IEEE Conference Proceedings*. doi: 10.1109/ICEET65156.2024.10913846.

[17] R. K. Ramakrishnan and J. J. Lekkala. Decentralized github management: Blockchain solution. Technical report, TechRxiv, 2025.

[18] Sameeruddin Shaik. Leveraging blockchain technology for a decentralized public key infrastructure: A blueprint for secure digital identity management. https://ieeexplore.ieee.org/document/11167581.

[19] S. Malipeddi. Enhancing phishing detection: The impact of visual warning designs and user engagement in email clients. In *Proceedings of the UNIfied Conference of DAMAS, IncoME VIII and TEPEN Conferences*. Springer, 2026. doi: 10.1007/978-3-031-95963-9_34.

[20] Rohit Puranik. Advancements in multi-party state channels for decentralized applications: A scalable approach to blockchain dispute resolution and execution. In *2025 International Conference on Information Technology, Electronics and Intelligent Communication Systems (ICITEICS)*, 2025. doi: 10.1109/ICITEICS64870.2025.11341106.

[21] Jitendra Gupta. Decentralized iot data trading with subscription model. In *IEEE Conference Proceedings*, 2026. URL https://ieeexplore.ieee.org/document/11447488/.

[22] Vinaykumar Chitukoori. Natural language processing: Sequence tagging for german semantic roles. *TechRxiv*, 2025. doi: 10.36227/techrxiv.176523100.04598096.

[23] Shujaatali Badami. Optimizing reinforcement learning in partially observable environments using compressed suffix memory algorithm. In *2024 IEEE 2nd International Conference on Electrical, Automation and Computer Engineering (ICEACE)*, pages 330–335, 2024. doi: 10.1109/ICEACE63551.2024.10899025.

[24] Priyanka Muppuri. Jump-attentive graph learning for camouflaged financial fraud in transaction networks.

[25] Tenaaz Syed. Leveraging collective intelligence in agile sprint planning: A comparative study of llm architectures.

[26] H. Hagui, A. Msolli, N. ben Henda, A. Helali, A. Gassoumi, T. P. Nguyen, and F. Hassen. A blockchain-based security system with light cryptography for user authentication security. *Multimedia Tools and Applications*, 83(17):52451–52480, 2024.

[27] A. Verma, A. Singh, P. Sethi, V. Jain, C. Chawla, A. Bhargava, and A. Gupta. Applications of data security and blockchain in smart city identity management. In *Handbook of Research on Data-Driven Mathematical Modeling in Smart Cities*, pages 154–174. IGI Global, 2023.

[28] L. Gudala, A. K. Reddy, A. K. R. Sadhu, and S. Venkataramanan. Leveraging biometric authentication and blockchain technology for enhanced security in identity and access management systems. *Journal of Artificial Intelligence Research*, 2(2):21–50, 2022.