DOI 10.20535/2411-1031.2024.12.2.315737
UDC 519.816, 004.056

VITALIY TSYGANOK,
MYKYTA SAVCHENKO,
ROMAN TSYHANOK

A UNIVERSAL TRANSACTION DELEGATION METHOD FOR DECENTRALIZED DECISION SUPPORT SYSTEMS

This study examines methods for decentralizing computation and storage to enhance the security of end systems, focusing on decision support systems as a use case. Common limitations of system decentralization are identified, and a new, universal transaction delegation method is proposed to simplify decentralized system usage. An overview of available transaction delegation methods in self-protected decentralized data platforms is provided, based on well-known projects using the Ethereum platform. Four popular delegation methods in decentralized networks are distinguished, with their advantages and disadvantages demonstrated through common solutions.

The research led to the implement of a universal transaction delegation method, independent of the decentralized program's signature standard. This method is realized as a web application on both the server and client sides and can be applied to any decentralized program or existing system supporting decentralized transaction delegation. The study also describes the architecture of a decision support system using this method, applied specifically to the expert subsystem to ensure decentralization and the integrity of expert input, making it impossible to tamper with once submitted.

Additionally, the economic model for the expert subsystem is reviewed, using real data. The findings of this study enable the construction of secure decentralized applications on decentralized data platforms, emphasizing usability and user-friendliness, and demonstrate an innovative application within a decision support system for expert knowledge collection.

Keywords: decentralized data platforms, delegated transactions, decision support systems, expert data, blockchain, Ethereum.

Introduction. Expert subsystems are a core component of decision support systems (DSS), significantly shaping prediction models. Expert evaluations are generally deemed reliable if they involve numerous experts or specialized expert groups [1]. To ensure unbiased input, various methods are implemented, such as anonymizing expert input and minimizing high-influence authorities' impact through distributed evaluations. However, guaranteeing data integrity in DSS often requires complex traditional security systems.

With the emergence of Distributed Ledger Technology (DLT) and decentralized data platforms (DDP), DSS can now leverage DDPs for secure, cost-effective data management compared to traditional methods. Today's DDPs, like blockchain, can operate on other synchronization technologies, such as hashgraph [2]. DDPs are maintained by numerous computing nodes in a secure network, making unauthorized access or tampering almost impossible. Key DDP properties include bandwidth, scalability, decentralization, and maintenance cost [3].

Modern DDPs like Ethereum, EOS, IOTA, Hedera Hashgraph, and Bitcoin serve various applications, from programmable cryptocurrencies and public registries to tokenization of assets [3]. Despite some incidents linked to third-party vulnerabilities, the decentralized platforms themselves have proven resilient. DDPs enable secure classic system upgrades, enhancing data protection and availability.

While DDPs offer benefits, they lack scalability and are often complex or costly for end users. Current DDPs cannot handle high-volume internet traffic, limiting their applications. Simplifying DDP-based applications is a crucial task, as user interaction with these systems remains challenging. This paper presents using of a universal approach [4] to designing DDP applications that enhance expert input security, mitigating tampering risks and fostering DSS trustworthiness.

1. The problem of multi-currency fees in decentralized data platforms

All DDPs require ongoing support from network participants motivated by incentives [4]. Common incentive schemes include:

– *Financial rewards*: In DDPs like Bitcoin and Ethereum, participants earn platform-specific assets as rewards, which also cover transaction fees. Specific reward models differ; some platforms inflate assets (e.g., 1% yearly) and distribute rewards among top participants.

– *Networking privileges*: IOTA, for example, requires users to validate others' transactions, allowing them to perform more transactions themselves.

– *Volunteer or institutional support*: Some public networks rely on volunteers or institutions, as seen with the Libra project, where organizations like Visa and PayPal contribute to network maintenance.

Most DDPs rely on financial rewards for network upkeep, with users paying fees in platform-specific cryptocurrencies [3]. This fee structure serves as a spam-prevention measure but complicates DDP access since users must purchase cryptocurrency and set up a wallet to start using DDP-based applications. This process limits user engagement, as illustrated in Figure 1.1.

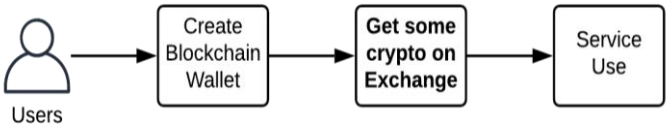

Figure 1.1 – User steps before using a decentralized application

Many users avoid DDP services due to complex onboarding, limiting adoption to those familiar with cryptocurrency. This complexity results in DDP-based applications either having limited user bases or simplifying functionalities, which undermines DDP advantages.

2. Approaches to simplifying decentralized data platform adoption

Decentralized applications (DApps) [3] require special software known as crypto wallets. Wallets, which may be standalone devices [5] or software [6], store users' private keys and help execute transactions within DDPs. Creating a wallet involves generating a key pair, with the private key stored securely offline [7]. Wallets may use algorithms like BIP32 [8] for generating multiple accounts from a single key, as shown in Figure 2.1.

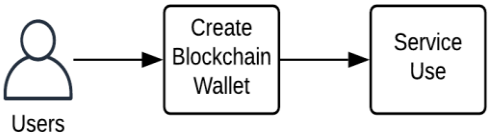

Figure 2.1 – Steps required before using a DApp without cryptocurrency purchase

To simplify user access, DDPs often employ transaction delegation. Here, a “delegate” handles the transactions, covering fees so the user avoids purchasing cryptocurrency [9], [10]. Below are the main transaction delegation methods.

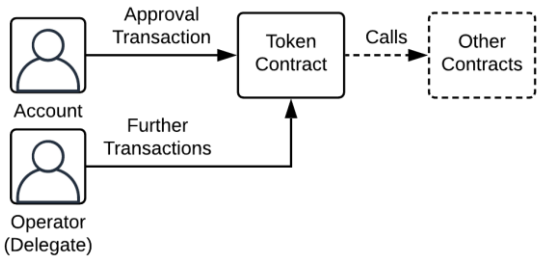

Figure 2.2 – Trusted transaction delegation approach scheme

2.1. Trusted transaction delegation. Trusted transaction delegation enables users to authorize another account to act on their behalf, as defined by standards like ERC777 in Ethereum [11]. This approach eases DApp use since the delegate manages transactions and fees. However, initial authorization still requires paying fees in platform-specific cryptocurrency. The scheme of a trusted delegated transaction approach is shown in Figure 2.2.

This delegation method benefits simplicity and standardization but may expose users to security risks due to the delegate's extensive permissions.

2.2. Transaction delegation with decentralized auxiliary identity programs. A more secure method employs auxiliary decentralized programs or identity contracts to delegate transactions. Users authorize transactions via signatures without paying fees, as shown in Figure 2.3. This method adds security by restricting the delegate's capabilities to signed actions.

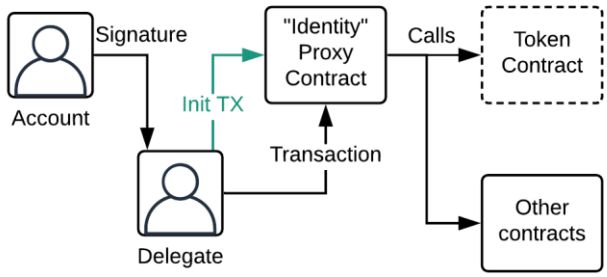

Figure 2.3 – Transaction delegation using auxiliary decentralized programs

While this method enhances security, it may be incompatible with certain DApps and requires an initial network fee for identity contract setup, which could expose DApps to spam attacks.

2.3. Transaction delegation without decentralized auxiliary identity programs

This method combines elements of both previous approaches. Users retain control over transactions through digital signatures without needing auxiliary programs. The method, illustrated in Figure 2.4, offers full control and is compatible with various DApps.

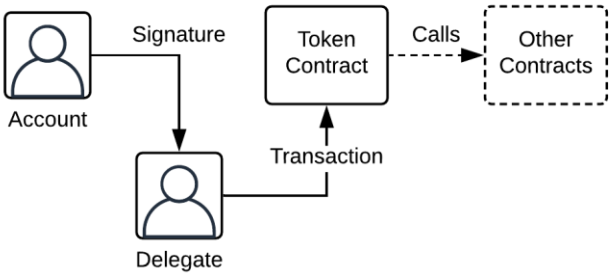

Figure 2.4 – Transaction delegation without auxiliary programs

Its advantages include reduced initialization requirements and compatibility with existing DApps. However, lack of standardization remains a limitation.

2.4. Transaction delegation at the decentralized data platform level. Implementing transaction delegation directly at the DDP level, as potentially planned for Ethereum 2.0 [12], could

streamline the delegation process. This method would allow delegates to cover transaction fees across multiple transactions without centralizing the DDP, as shown in Figure 2.5.

While this approach would encompass all previous delegation benefits, it faces challenges in standardization and adoption across platforms.

3. The universal method of transaction delegation

All the above transaction delegation methods have their pros and cons, with none standing out as the best. To assess which approach may be the most suitable for further development, a review of popular decentralized applications (DApps) on Ethereum that successfully completed an ICO was conducted, including projects such as Binance, DreamTeam, Loom Network, Stratis, Maker DAO, OmiseGo, Basic Attention Token, 0x, and Golem. Key trends were identified:

– Most DApps use their own token on top of the decentralized data platform (DDP) to pay for their services [13].

– DApps are generally limited to an experienced cryptocurrency audience, who often avoid delegated transactions and instead pay with the DDP currency alongside the service token [14].

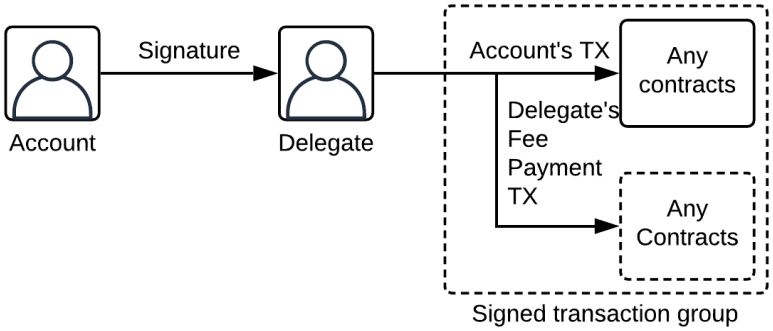

Figure 2.5 – Transaction delegation scheme at the DDP level

– Private DApps that use transaction delegation create specialized, non-interoperable systems that work only with their own tokens.

– Transaction delegation tends to be centralized, lacking an open ecosystem (market) for delegated transactions. DApp developers generally manage delegated accounts without financial interest for other organizations.

– There is no common standard for building DApps with transaction delegation.

In summary, most DApps build their own delegation ecosystems, embedding delegated functions within the decentralized token program. However, the absence of a unified approach or standard remains a problem.

The universal method of transaction delegation proposed here aims to standardize DApp development without requiring a single standard at the DDP level. This approach allows a single backend service system for any DApp with transaction delegation, supporting both new and existing programs.

This method provides a reference for token development, supporting multiple implementations and ensuring security for DApp users. The universal method involves three main components:

– A flexible approach to building decentralized programs, allowing developers to choose any transaction delegation standard.

– A backend service component compatible with any DApp and transaction delegation approach.

– A universal, embeddable client-side UI as a web widget, highly configurable for executing delegated transactions.

3.1. The approach to decentralized program creation. This universal transaction delegation method operates at the DDP level, embedding delegated functions within the token program rather than relying on an intermediate (identity) contract. Delegated functions are integrated directly into the decentralized token program, which also handles transaction fees. Figure 3.1 shows user interaction with the DApp without intermediate contracts.

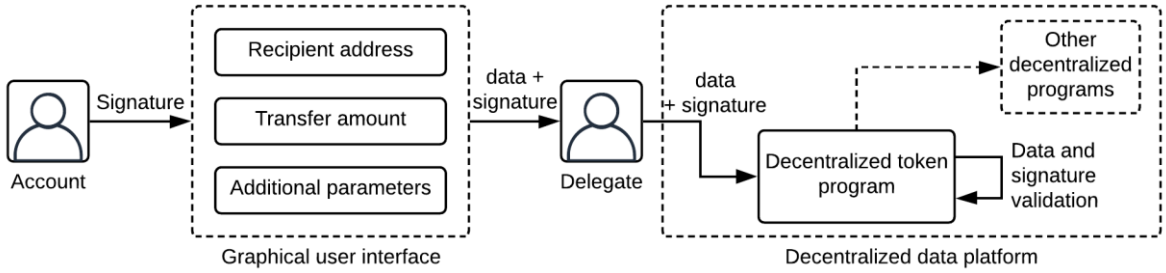

Figure 3.1 – the user interaction with the decentralized program without intermediate (identity) decentralized programs

In this approach, a delegated transaction is executed as follows:

1. The user enters data or receives pre-prepared data in the graphical UI.
2. The user signs the data and additional transaction-specific parameters (such as commission, deadline, and transaction ID).
3. The user sends the signed data to the delegate.
4. The delegate sends the transaction to the DDP, paying the commission.
5. The token program verifies the signature and data, then executes the transaction if valid.

A decentralized token program can be built using any standard, such as ERC20 or ERC721 [15]. However, to use this universal method effectively, the program must meet the following criteria:

– Every normal function of the decentralized program should have a "delegated" counterpart, which performs the same action without depending on the calling account. Instead, it relies on a digital signature to identify the account. For example, if a token program includes a transfer function, a transferViaSignature function should be added, allowing transactions to be executed by a delegate. Alternatively, a single delegate function could handle all other functions via electronic signatures.

– If possible, the delegated function should support multiple signature standards to ensure compatibility and resilience.

– The delegated function should accept additional parameters to enhance security and limit misuse. Recommended parameters include transaction ID, deadline, and a fee recipient account. All parameters should be signed and verified by the token program.

– A proxy-call function may also be implemented, along with a corresponding delegated function, often called approveAndCall. This allows other DApps to interact with tokens in a single transaction.

The following describes the recommended approach to implementing delegated functions in a decentralized Ethereum data platform, specifically for an ERC20 token and its transfer function. This approach simplifies the DApp by removing the need for a separate currency for transaction fees. The method can be applied to any DDP unless the platform offers another transaction delegation option.

The ERC20 transfer function is designed to move tokens from one account to another. In this model, a transferViaSignature function is added to perform the same operation through delegation, allowing any account to initiate a transfer on behalf of the account that signed the transaction. Figure 3.2 demonstrates the process and outcomes of the transfer and transferViaSignature functions. Notably, transferViaSignature can include an optional commission to offset the delegate's expenses.

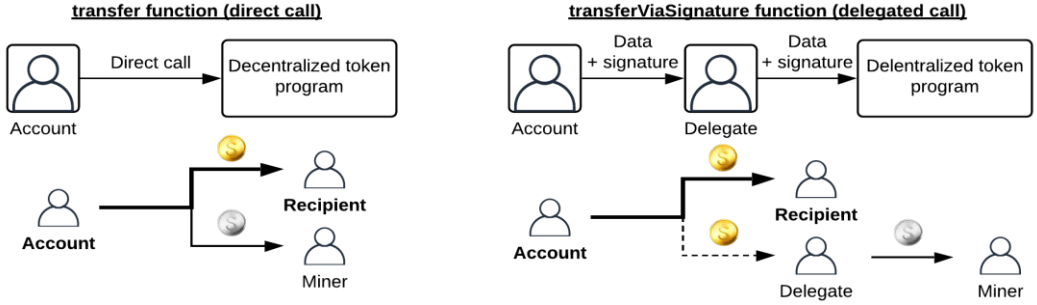

Figure 3.2 – the scheme of calls to transfer and transferViaSignature functions and function results

In a delegated transaction, the account covers transaction fees with the token itself, while the delegate pays the DDP's commission. The delegate can exchange tokens to cover their expenses in the background, eliminating the need for the user to hold multiple currencies (e.g., Ether on Ethereum). Commissions may be omitted, but in this model, a balanced economic structure rewards the delegate for facilitating the transaction. The transferViaSignature function includes the same parameters as the transfer function, along with additional ones for enhanced security.

Figure 3.3 illustrates the recommended parameters:

– from – the sender's account (optional), distinct from the calling account and can be retrieved from the digital signature.

– fee – the commission amount the sender pays to the feeRecipient.

```
contract ERC20 {
  function transfer(to, value) external;
  function transferViaSignature(from, to, value, fee, feeRecipient, deadline, sigId, sig, sigStd) external;
}
```

Figure 3.3 – recommended parameters of the functions of a decentralized token program

– feeRecipient – the account receiving the commission; should not default to the calling account to avoid race conditions.

– deadline – the delegated transaction execution limit, allowing users to control transaction validity and re-sign if necessary.

– sigId – a unique identifier for the transaction, verified for uniqueness to prevent repeated execution.

– sig – the digital signature of all parameters, certifying the action.

– sigStd – a signature standard identifier for interoperability across standards.

These additional parameters restrict the delegate's ability to manipulate the transaction, ensuring that only the defined transaction is executed as intended.

The universal method introduces a degree of centralization since only specific entities can offer delegation services. However, this limited centralization is solely to simplify the DApp for users. To maintain decentralization, developers are encouraged to write delegated functions as supplementary features, keeping the original functions available for direct user transactions.

3.2. The approach to transaction delegation support service creation. Developing a decentralized program for delegated transactions requires creating an automated server-side workflow for conducting transactions on a decentralized data platform. The support service should handle the following functions:

– Supporting real-time delegated transactions, ensuring that the transaction is sent to the network and stored (mined).

– Collecting data required for delegated transactions.

– Calculating the reasonable transaction fee based on network load and asset exchange rates.

– Validating and verifying user data.

– Allowing users to track delegated transactions.

The task was to develop an auxiliary server system that would support these functions, compatible with any decentralized system, including those supporting transaction delegations. This technical challenge involves anticipating:

– Protection against various attacks, such as spam, duplication of transactions, and unauthorized access.

– Sequential execution of delegated transactions, even when requests can be parallel (for some platforms).

– Management of transaction parameters, ensuring profitable transaction fees.

– Serving multiple user requests simultaneously.

– Ensuring system versatility for any decentralized platform supporting transaction delegation.

For example, in the Ethereum decentralized data platform, the interaction of system components is illustrated in Figure 3.4.

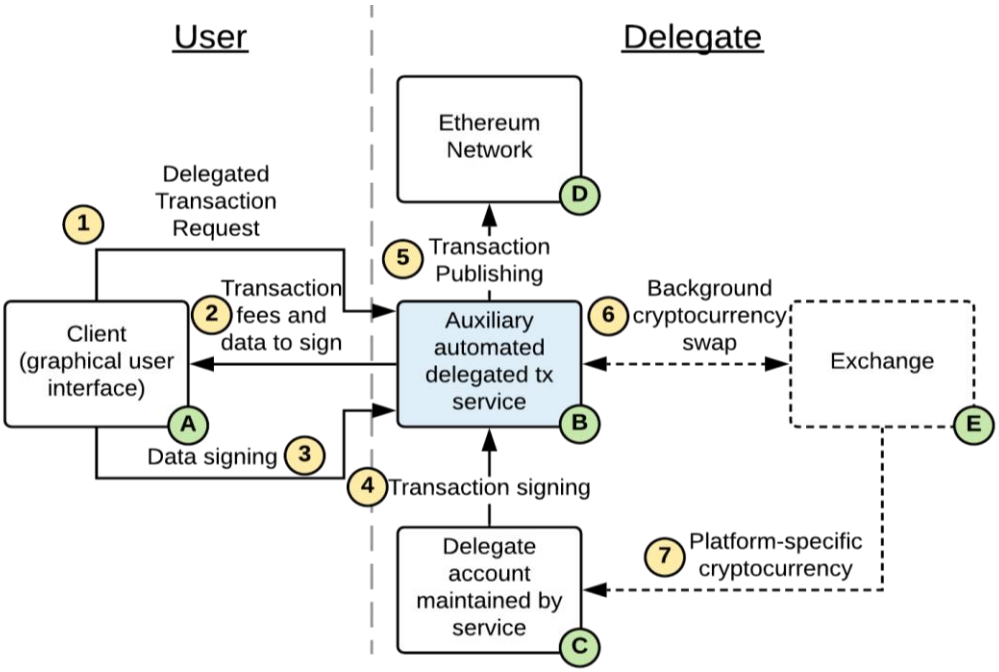

Figure 3.4 – Interaction between the delegation system components and the auxiliary server part

The auxiliary system includes five key components:

– A (client interface): The graphical interface for user interaction.

– B (automated server): The delegation server, which clients choose based on the best transaction rate.

– C (delegate account): Stores the private key and signs transactions.

– D (decentralized network): Supported by the platform (e.g., Ethereum).

– E (exchange module): Handles automatic cryptocurrency exchange to pay fees in platform-specific cryptocurrency.

The system may not include the exchange module in some configurations, particularly in private systems where users are not charged transaction fees. Public transaction delegation requires a fee, which the delegate may adjust for profit. Fees are converted into platform-specific cryptocurrency to cover transaction costs.

3.3. User's interaction with the delegation system. The user interaction with the system and the interaction between the system components is illustrated in Figure 3.5.

The automated steps for user interaction are as follows:

1. Delegated transaction request: The client prepares the transaction parameters and requests approval from the Transaction Delegation Support Service.

2. Delegate's response: Using REST API, the delegate informs the client about:

– Feasibility of the transaction.

– The fee in the user’s token (calculated using an algorithm).

– Signature data for the user, offering options for different signature standards.

– Additional details, such as transaction ID and estimated time.

3. User signature and transaction confirmation: The client selects the signature method, signs the data, and sends it to the delegate for execution.

4. Transaction signature by delegate: The delegate signs the transaction and submits it to the network, including user parameters, delegate-defined options, and user signature details.

5. Transaction submission to the decentralized network: Once validated, the transaction is sent to the network. After mining, the delegate notifies the client of success or failure, and the client checks the status.

6. Getting the result: The client continuously polls the delegate or checks the transaction status independently in the decentralized network.

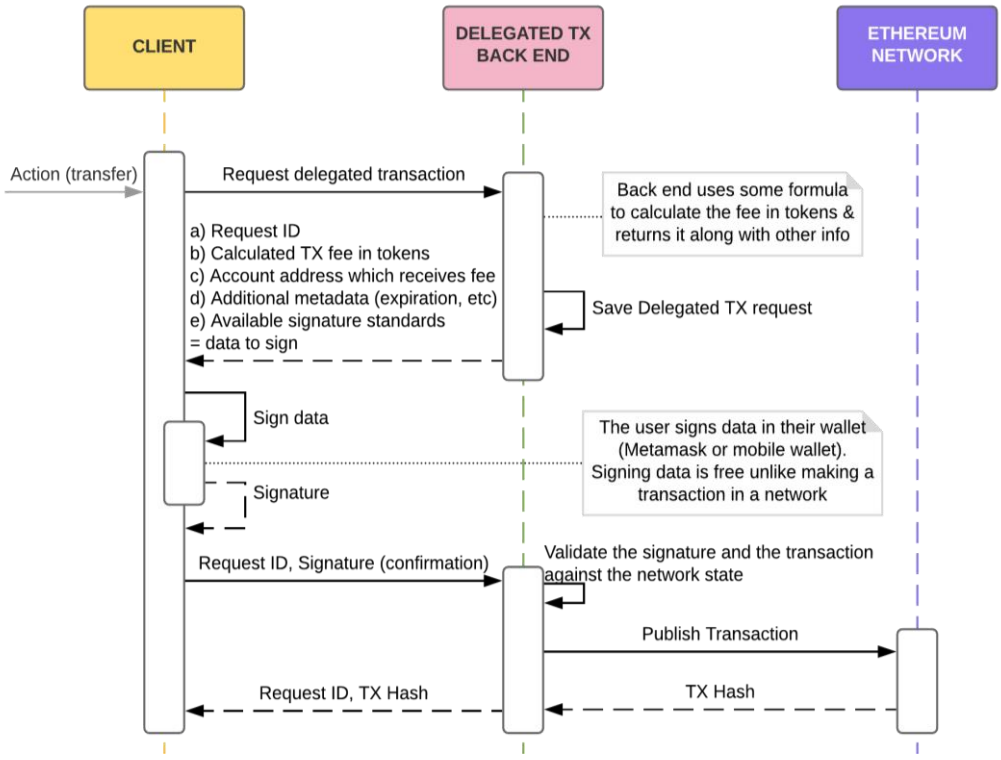

Figure 3.5 – Sequence of user interaction with the delegate and decentralized network

This method automates most interactions, with the user only needing to sign the transaction. The signature ensures transaction integrity, preventing unauthorized changes by the delegate.

4. The model of decentralized decision support system

Decision Support Systems (DSSs) are used to generate recommendations based on facts and expert input [16-18]. However, centralized storage of expert data poses risks of unauthorized access and modification. A decentralized system mitigates these risks, ensuring reliable input data from experts, which is crucial for DSS credibility.

4.1. Decentralization of the expert subsystem. The decentralized model is illustrated in Figure 4.1, where expert data is stored in a decentralized registry, making it auditable and authentic.

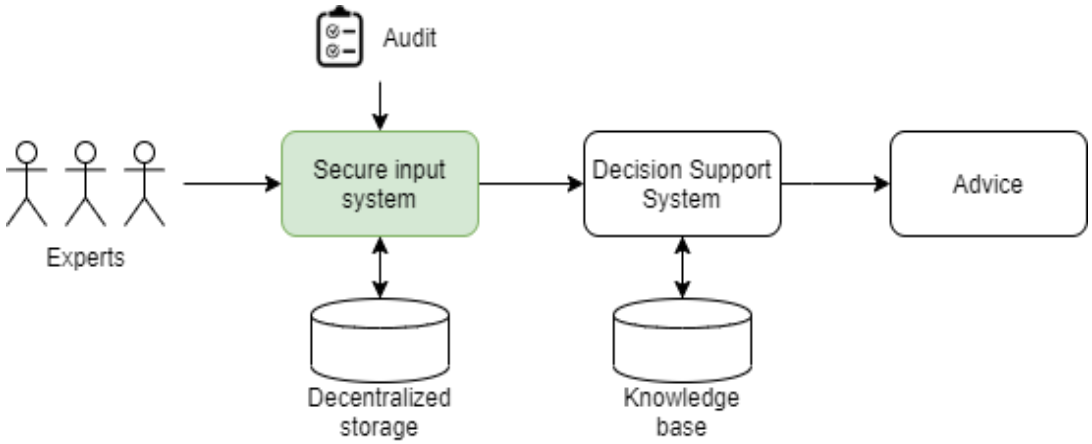

Figure 4.1 – DSS model with a decentralized expert subsystem

The DSS cycle consists of these steps:

1. Setup phase: Experts obtain decentralized accounts (wallets), replacing traditional login systems.
2. Funding phase: Experts fund their decentralized accounts to pay transaction fees. Delegated transactions can eliminate this step.
3. Authorization and data input: Expert-submitted data is stored securely in the decentralized registry, minimizing data volume to reduce costs.
4. Data submission and signature: Experts sign data and submit it to the decentralized network for verification.
5. Data processing: The DSS processes the expert data for further use.

4.2. The use of delegated transactions in the expert subsystem. Delegated transactions streamline expert interactions, as shown in Figure 4.2. In this model, experts do not manage transaction fees directly. Instead, a system operator funds the decentralized account to cover transaction costs, reducing the expert's financial burden. The system operator funds the account once, for example, with $50, which covers multiple expert operations until the next top-up is needed.

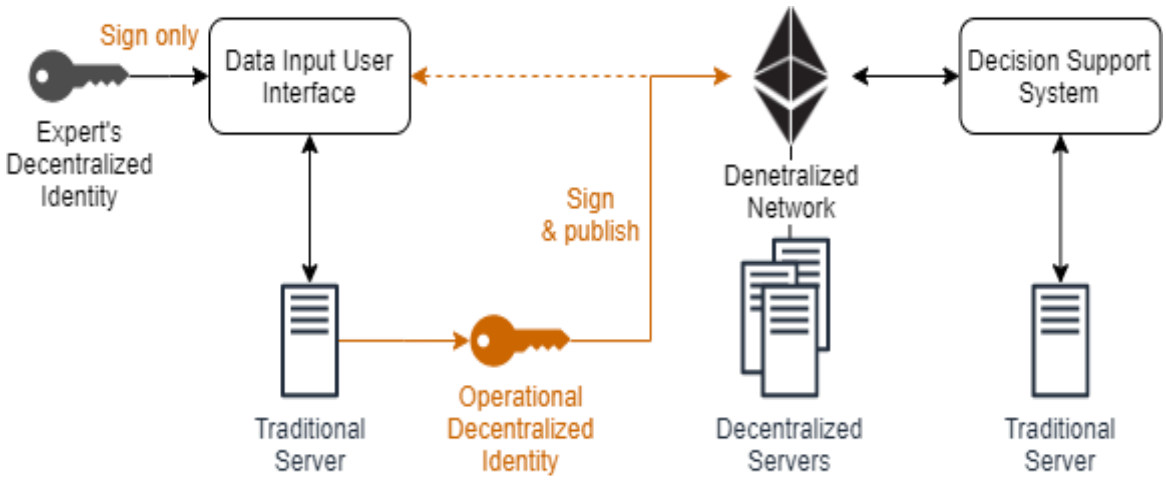

Figure 4.2 – Decentralized expert subsystem model using delegated transactions

By utilizing transaction delegation, experts can focus on their work without worrying about transaction fees, making the system more efficient and user-friendly.

4.3. Ways to compactify the data. Unlike free data read operations, transaction fees are incurred for every byte written or changed in a public decentralized registry (excluding test networks). Thus, minimizing the data recorded in the decentralized registry is crucial.

In Ethereum, transaction costs are calculated based on the integer G (1), known as "gas". It represents the sum of the costs of all executed program instructions g_i, each with a predefined value. Some operations, such as deleting data, have a negative value, but the sum of these cannot exceed half of the positive amount. Each transaction also has a base cost $g_0 = 21000$ added to G, and the total cost is limited by the block size $G_{\max} = 10000000$ (as of early 2020). The transaction cost is thus:

$$G = g_0 + \sum_{i=1}^{N} g_i [g_i > 0] - \min\left\{ -\sum_{i=1}^{N} g_i [g_i < 0], \left(g_0 + \sum_{i=1}^{N} g_i [g_i > 0] \right) \div 2 \right\}, G \le G_{\max}. \quad (1)$$

The final transaction cost in cryptocurrency X (Ether) is defined as $E = G' P$, where P is the "gas price", set by the transaction signer. While P could theoretically be zero, miners would have no financial incentive to include such a transaction. Therefore, selecting the appropriate P for the current network conditions is necessary.

The most "valuable" operations in Ethereum are those that save data to the decentralized registry. Writing a new value (256 bits) costs $g = 20000$, and modifying an existing value costs $g = 5000$. For more complex data structures, additional memory slots may be needed, which are priced similarly [19].

To reduce the transaction cost, one can minimize G (by reducing the number of transactions and increasing data deletion) and P (by performing transactions during low network load). Below, we explore methods for compacting data and evaluate their impact on system costs in a real public decentralized network.

Compacting expert records for a decentralized registry

Let's assume that expert evaluations are stored in the most compact form in real-time. For instance, the binary expert assessments $a_1, a_2, \text{K}\ a_n$ (obtained by pairwise comparisons) are written as an array of bits $b_1, b_2, \text{K}\ b_n$ where each bit corresponds to a binary evaluation of two objects by a specific criterion. The number of pairwise comparisons is $n = k' (k - 1), 2$, where k is the number of objects being compared [18].

The storage cost for 256 bits of information in the decentralized registry is $G = g_0 + g_{sstore}$, where $g_{sstore} = 20000$ is the cost of writing 256 bits. Assuming that each expert session generates an average of $b_{tx} = 1' 10^3$ bits of data, the cost function for storing data is:

$$f(b) = g_0 + [b \div 256] \times g_{sstore} + [b \div b_{tx}] \times (g_0 + g_{sstore}) \text{K} . \quad (2)$$

As shown in Figure 4.3, the relationship between the amount of data stored and the transaction value is nearly linear, with some exceptions where data is split into multiple transactions due to individual submissions.

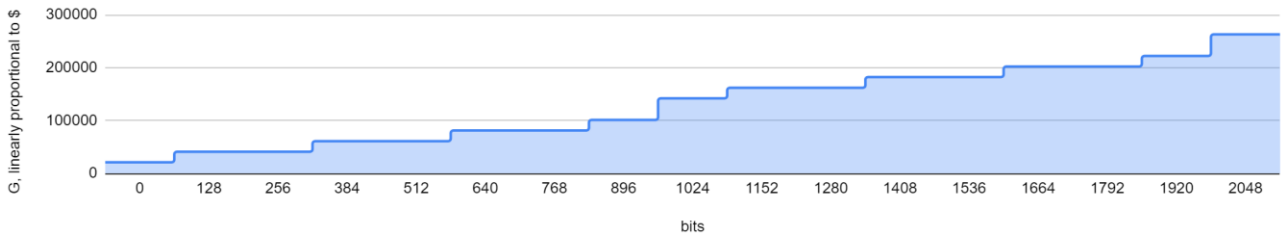

Figure 4.3 – Relation between G and the number of bits of data stored in the decentralized registry

Overwriting outdated data

A more cost-effective approach is to overwrite outdated data rather than appending new data. In Ethereum, overwriting 256 bits of pre-existing data costs $g_{sstore} = 5000$, significantly reducing the transaction price by up to 61.3% compared to recording new data.

However, this method imposes restrictions on retrieving outdated data. While decentralized programs do not provide direct access to overwritten data, historical information can still be recovered using transaction or block IDs, stored outside the registry. Blockchain technology allows recovery of overwritten data through non-standard interfaces when needed.

Transactions grouping

Grouping multiple expert transactions is possible if the data does not need to be recorded in real-time, allowing for delayed entries in the decentralized registry. Implementing this requires additional modules and precautions to ensure the expert's data is included. One way to reduce costs is by using delegated transactions, which allow the transaction to be submitted by an administrator.

Grouping transactions without delegated transactions can result in up to 16.54% cost savings, reaching a total of 77.84%. However, there is a limit on the number of transactions that can be grouped, as Ethereum has a block size limit of $G = 10\,000\,000$. If grouped transactions exceed the block size (e.g., 10.5 KB), they must be split into separate groups.

Impact of delegated transactions on the total cost of the system

Delegated transactions add a small additional cost to each transaction, but this becomes significant when transactions are not grouped. Experimental results show that performing the minimal data-recording transaction with the method described here adds approximately

$g_{0\,\text{extra}} \approx g_{0\,\text{extra min}} \approx 51000$ in additional costs. However, the signature validation logic does not significantly depend on the data size, making this additional cost negligible in calculations.

Figure 4.4 illustrates the relationship between G and the number of bits written to the decentralized registry for all optimization methods:

- Real-time data recording (blue).
- Overwriting outdated data (red).
- Deferred transaction grouping (purple).
- Grouping with delegated transactions (green).
- Grouping with delegated transactions in a more optimized system (yellow).

From Figure 4.4, it is clear that using delegated transactions can reduce system costs by over 50% when combined with transaction grouping. However, implementing delegated transactions increases the minimal system cost by only 18.75% compared to the most optimized method without delegation.

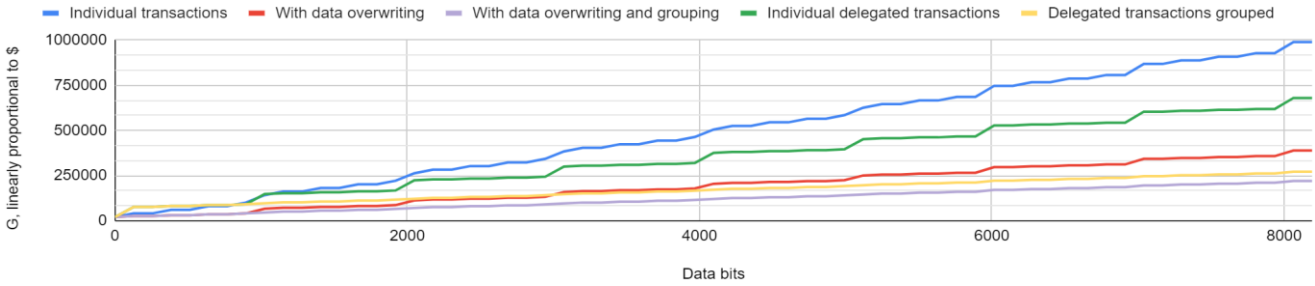

Figure 4.4 – Dependence of value G on the number of bits of recorded information for all methods of data compactification

4.4. Estimating the real cost of system operation. The cost evaluation of the expert system's operation was conducted on the Ethereum network, using a system that compacted data by overwriting outdated entries without grouping transactions. Over a three-month period, an average of two delegated transactions were conducted daily, each modifying 768 bits in the decentralized registry. Approximately 90% of these transactions simply overwrote existing data. The set gas price P reflected the real-time network load, without attempting cost reductions through deferred transactions.

The study confirms the transaction delegation system's readiness for real-world deployment, though some optimizations for cost reduction – such as transaction grouping and deferral to low-traffic times – were not implemented in this phase.

Based on these results, we assessed cost savings achievable by:

1. Conducting transactions during low network load, which could reduce the average gas price $P_{avg} = 1.01' 10^9$, yielding a 74% 74% cost savings;

2. Using data compactification with transaction grouping: 95 transactions could be consolidated into two larger transactions, with an additional 74% reduction in costs. This value can be calculated by substituting the aggregate values in formula (2) for $g_0 = g_{0\,\text{extra min}} = 51000$;

3. Combining both strategies would yield a maximum cost reduction of 92.2%.

Thus, the estimated minimal cost to write $b = 71.25\,\text{KB}$ of data in two delegated transactions would be approximately $0.40. This estimate depends on cryptocurrency price fluctuations and network demand [20], and doesn't include additional logic beyond basic data writing [21]. In some cases, high network fees could delay transactions by months if budget constraints prevent timely processing.

Conclusions. A universal transaction delegation method based on research into decentralized data platforms and applications is presented. This method standardizes the backend and client side without requiring standardization of the decentralized application itself, making it applicable to both

new and existing systems. Our optimizations applied to expert subsystems of decision support systems can reduce typical operating costs by up to 92.2%.

The transaction delegation system developed here simplifies the user experience and eases developer implementation efforts. Beyond decision support systems, the method has been applied to real-world business applications, including projects like Decash and Hoprnet. During testing, a project with nearly 2000 users demonstrated the system's scalability and readiness for decentralized applications on Ethereum.

Future uses of these solutions span secure decentralized applications with user-friendly interfaces, especially for systems needing high security and data integrity, such as medical and financial applications. This method provides a foundation for creating decentralized applications that balance security with affordability, supporting the wider adoption of blockchain technology.

REFERENCES

[1] V.V. Tsyganok, S.V. Kadenko, and O.V. Andriichuk, "Significance of expert competence consideration in group decision making using AHP", *International Journal of Production Research.*, vol. 50, no. 17, pp. 4785-4792, 2012. doi: https://doi.org/10.1080/00207543.2012.657967.

[2] D. Mookherjee, "Decentralization, hierarchies, and incentives: A mechanism design perspective", *Journal of Economic Literature*, vol. 44, no. 2, pp. 367-390, 2006. doi: https://doi.org/10.1257/jel.44.2.367.

[3] S. Raval, *Decentralized Applications: Harnessing Bitcoin's Blockchain Technology*. Cambridge, MA, USA: O'Reilly Media, Inc., 2016.

[4] N. Savchenko, V. Tsyganok, and O. Andriichuk, "A Cost-Effective Approach to Securing Systems through Partial Decentralization", *Information & Security*: *An International Journal*, vol. 47, no. 1, pp. 109-121, 2020. doi: https://doi.org/10.11610/isij.4707.

[5] M. Arapinis, A. Gkaniatsou, D. Karakostas, and A. Kiayias, "A Formal Treatment of Hardware Wallets", in *Financial Cryptography and Data Security. FC 2019*, I. Goldberg, T. Moore, Eds., vol. 11598. Cham: Springer, 2019, pp. 426-445. doi: https://doi.org/10.1007/978-3-030-32101-7_26.

[6] M. Pustišek, and A. Kos, "Approaches to front-end IoT application development for the Ethereum blockchain", *Procedia Computer Science*, vol. 129, pp. 410-419, 2018. doi: https://doi.org/10.1016/j.procs.2018.03.017.

[7] H. Rezaeighaleh, and C.C. Zou, "New Secure Approach to Backup Cryptocurrency Wallets", in *Proc. 2019 IEEE Global Communications Conference (GLOBECOM)*, Waikoloa, HI, USA, 2019, pp. 1-6. doi: https://doi.org/10.1109/GLOBECOM38437.2019.9014007.

[8] D. Khovratovich, and J. Law, "BIP32-Ed25519: Hierarchical Deterministic Keys over a Non-linear Keyspace", in *Proc. 2017 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, April, 2017, pp. 27-31. [Online]. Available: https://input-output-hk.github.io/adrestia/static/Ed25519_BIP.pdf. Accessed on: May 26, 2024.

[9] S. Cho, S.Y. Park, and S.R. Lee, "Blockchain consensus rule based dynamic blind voting for non-dependency transaction", *International Journal of Grid and Distributed Computing*, vol. 10, no. 12, pp.93-106, 2017. doi: https://doi.org/10.14257/ijgdc.2017.10.12.09.

[10] A. Ouaddah, A.A. Elkalam, and A.A. Ouahman, "Towards a novel privacy-preserving access control model based on blockchain technology in IoT", in *Europe and MENA Cooperation Advances in Information and Communication Technologies*. New York, USA: Springer, 2017, pp. 523-533. doi: https://doi.org/10.1007/978-3-319-46568-5_53.

[11] M. Mulders, "Comparing ERC20, ERC223, and the new Ethereum ERC777 token standard". [Online]. Available: https://www.cointelligence.com/content/comparison-erc20-erc223-new-ethereum-erc777-token-standard. Accessed on: May 27, 2024.

[12] S.J. Wels, “Guaranteed-TX: The exploration of a guaranteed cross-shard transaction execution protocol for Ethereum 2.0”, Master's thesis, University of Twente, Enschede, Netherlands, 2019. [Online]. Available: https://purl.utwente.nl/essays/79884. Accessed on: May 27, 2024.
[13] J.Y. Lee, “A decentralized token economy: How blockchain and cryptocurrency can revolutionize business”, *Business Horizons*, vol. 62, no. 6, pp. 773-784, 2019. doi: https://doi.org/10.1016/j.bushor.2019.08.003.
[14] M. Iansiti, and K.R. Lakhani, “The truth about blockchain”, *Harvard Business Review*, vol. 95, no. 1, pp. 118-127, 2017. [Online]. Available: https://www.researchgate.net/publication/341913793_The_Truth_About_Blockchain. Accessed on: Sep. 27, 2024.
[15] M. Kim, B. Hilton, Z. Burks, and J. Reyes, “Integrating Blockchain, Smart Contract-Tokens, and IoT to Design a Food Traceability Solution”, in *Proc. 2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, Vancouver, 2018, pp. 335-340. doi: https://doi.org/10.1109/IEMCON.2018.8615007.
[16] T.L. Saaty, *Principia Mathematica Decernendi - Mathematical principles of decision making - Generalization of the Analytic Network Process to neural firing and synthesis*, Pittsburg, PA, USA: RWS Publications, 2010.
[17] V. Tsyganok, S. Kadenko, O. Andriychuk, and P. Roik, “Usage of multicriteria decision-making support arsenal for strategic planning in environmental protection sphere”, *Journal of Multi-Criteria Decision Analysis*. vol. 24, pp. 227-238, 2017. doi: https://doi.org/10.1002/mcda.1616.
[18] V.G. Totsenko, and V.V. Tsyganok, “Method of paired comparisons using feedback with expert”, *Journal of Automation and Information Sciences*, vol. 31, no. 7-9, pp. 86-96, 1999. doi: https://doi.org/10.1615/JAutomatInfScien.v31.i7-9.480.
[19] P.R. Manoj, *Ethereum Cookbook: Over 100 Recipes Covering Ethereum-Based Tokens, Games, Wallets, Smart Contracts, Protocols, and Dapps*, Birmingham, UK: Packt Publishing Ltd., 2018.
[20] K. Wołk, “Advanced social media sentiment analysis for short-term cryptocurrency price prediction”, *Expert Systems*, vol. 37, no. 2, 2020. doi: https://doi.org/10.1111/exsy.12493.
[21] V.V. Tsyganok, and S.V. Kadenko, “On sufficiency of the consistency level of group ordinal estimates”, *Journal of Automation and Information Sciences*, vol. 42, no. 8, pp. 42-47, 2010. doi: https://doi.org/10.1615/JAutomatInfScien.v42.i8.50.

The article was received 14.07.2024.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

[1] V.V. Tsyganok, S.V. Kadenko, and O.V. Andriichuk, “Significance of expert competence consideration in group decision making using AHP”, *International Journal of Production Research.*, vol. 50, no. 17, pp. 4785-4792, 2012. doi: https://doi.org/10.1080/00207543.2012.657967.
[2] D. Mookherjee, “Decentralization, hierarchies, and incentives: A mechanism design perspective”, *Journal of Economic Literature*, vol. 44, no. 2, pp. 367-390, 2006. doi: https://doi.org/10.1257/jel.44.2.367.
[3] S. Raval, *Decentralized Applications: Harnessing Bitcoin's Blockchain Technology*. Cambridge, MA, USA: O'Reilly Media, Inc., 2016.
[4] N. Savchenko, V. Tsyganok, and O. Andriichuk, “A Cost-Effective Approach to Securing Systems through Partial Decentralization”, *Information & Security*: *An International Journal*, vol. 47, no. 1, pp. 109-121, 2020. doi: https://doi.org/10.11610/isij.4707.
[5] M. Arapinis, A. Gkaniatsou, D. Karakostas, and A. Kiayias, “A Formal Treatment of Hardware Wallets”, in *Financial Cryptography and Data Security. FC 2019*, I. Goldberg, T. Moore, Eds.,

vol. 11598. Cham: Springer, 2019, pp. 426-445. doi: https://doi.org/10.1007/978-3-030-32101-7_26.

[6] M. Pustišek, and A. Kos, "Approaches to front-end IoT application development for the Ethereum blockchain", *Procedia Computer Science*, vol. 129, pp. 410-419, 2018. doi: https://doi.org/10.1016/j.procs.2018.03.017.

[7] H. Rezaeighaleh, and C.C. Zou, "New Secure Approach to Backup Cryptocurrency Wallets", in *Proc. 2019 IEEE Global Communications Conference (GLOBECOM)*, Waikoloa, HI, USA, 2019, pp. 1-6. doi: https://doi.org/10.1109/GLOBECOM38437.2019.9014007.

[8] D. Khovratovich, and J. Law, "BIP32-Ed25519: Hierarchical Deterministic Keys over a Non-linear Keyspace", in *Proc. 2017 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, April, 2017, pp. 27-31. [Online]. Available: https://input-output-hk.github.io/adrestia/static/Ed25519_BIP.pdf. Accessed on: May 26, 2024.

[9] S. Cho, S.Y. Park, and S.R. Lee, "Blockchain consensus rule based dynamic blind voting for non-dependency transaction", *International Journal of Grid and Distributed Computing*, vol. 10, no. 12, pp.93-106, 2017. doi: https://doi.org/10.14257/ijgdc.2017.10.12.09.

[10] A. Ouaddah, A.A. Elkalam, and A.A. Ouahman, "Towards a novel privacy-preserving access control model based on blockchain technology in IoT", in *Europe and MENA Cooperation Advances in Information and Communication Technologies*. New York, USA: Springer, 2017, pp. 523-533. doi: https://doi.org/10.1007/978-3-319-46568-5_53.

[11] M. Mulders, "Comparing ERC20, ERC223, and the new Ethereum ERC777 token standard". [Online]. Available: https://www.cointelligence.com/content/comparison-erc20-erc223-new-ethereum-erc777-token-standard. Accessed on: May 27, 2024.

[12] S.J. Wels, "Guaranteed-TX: The exploration of a guaranteed cross-shard transaction execution protocol for Ethereum 2.0", Master's thesis, University of Twente, Enschede, Netherlands, 2019. [Online]. Available: https://purl.utwente.nl/essays/79884. Accessed on: May 27, 2024.

[13] J.Y. Lee, "A decentralized token economy: How blockchain and cryptocurrency can revolutionize business", *Business Horizons*, vol. 62, no. 6, pp. 773-784, 2019. doi: https://doi.org/10.1016/j.bushor.2019.08.003.

[14] M. Iansiti, and K.R. Lakhani, "The truth about blockchain", *Harvard Business Review*, vol. 95, no. 1, pp. 118-127, 2017. [Online]. Available: https://www.researchgate.net/publication/341913793_The_Truth_About_Blockchain. Accessed on: Sep. 27, 2024.

[15] M. Kim, B. Hilton, Z. Burks, and J. Reyes, "Integrating Blockchain, Smart Contract-Tokens, and IoT to Design a Food Traceability Solution", in *Proc. 2018 IEEE 9th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, Vancouver, 2018, pp. 335-340. doi: https://doi.org/10.1109/IEMCON.2018.8615007.

[16] T.L. Saaty, *Principia Mathematica Decernendi - Mathematical principles of decision making - Generalization of the Analytic Network Process to neural firing and synthesis*, Pittsburg, PA, USA: RWS Publications, 2010.

[17] V. Tsyganok, S. Kadenko, O. Andriychuk, and P. Roik, "Usage of multicriteria decision-making support arsenal for strategic planning in environmental protection sphere", *Journal of Multi-Criteria Decision Analysis*. vol. 24, pp. 227-238, 2017. doi: https://doi.org/10.1002/mcda.1616.

[18] V.G. Totsenko, and V.V. Tsyganok, "Method of paired comparisons using feedback with expert", *Journal of Automation and Information Sciences*, vol. 31, no. 7-9, pp. 86-96, 1999. doi: https://doi.org/10.1615/JAutomatInfScien.v31.i7-9.480.

[19] P.R. Manoj, *Ethereum Cookbook: Over 100 Recipes Covering Ethereum-Based Tokens, Games, Wallets, Smart Contracts, Protocols, and Dapps*, Birmingham, UK: Packt Publishing Ltd., 2018.

[20] K. Wołk, "Advanced social media sentiment analysis for short-term cryptocurrency price prediction", *Expert Systems*, vol. 37, no. 2, 2020. doi: https://doi.org/10.1111/exsy.12493.

[21] V.V. Tsyganok, and S.V. Kadenko, “On sufficiency of the consistency level of group ordinal estimates”, *Journal of Automation and Information Sciences,* vol. 42, no. 8, pp. 42-47, 2010. doi: https://doi.org/10.1615/JAutomatInfScien.v42.i8.50.

ВІТАЛІЙ ЦИГАНОК,
МИКИТА САВЧЕНКО,
РОМАН ЦИГАНОК

УНІВЕРСАЛЬНИЙ МЕТОД ДЕЛЕГУВАННЯ ТРАНЗАКЦІЙ ДЛЯ ДЕЦЕНТРАЛІЗОВАНИХ СИСТЕМ ПІДТРИМКИ ПРИЙНЯТТЯ РІШЕНЬ

У цьому дослідженні розглядаються методи децентралізації обчислень і зберігання даних для підвищення безпеки систем, зосереджуючись на системах підтримки прийняття рішень як на прикладі використання. Визначено загальні обмеження децентралізації систем та запропоновано новий універсальний метод делегування транзакцій для спрощення використання децентралізованих систем. Наведено огляд доступних методів делегування транзакцій у самозахищених децентралізованих платформах даних на основі відомих проектів, що використовують платформу Ethereum. Виділено чотири популярні методи делегування в децентралізованих мережах, продемонстровано їх переваги та недоліки на прикладі поширених рішень.

В результаті дослідження було реалізовано універсальний метод делегування транзакцій, незалежний від стандарту підпису децентралізованої програми. Цей метод реалізований у вигляді веб-додатку як на стороні сервера, так і на стороні клієнта і може бути застосований до будь-якої децентралізованої програми або існуючої системи, що підтримує децентралізоване делегування транзакцій. У дослідженні також описано архітектуру системи підтримки прийняття рішень з використанням цього методу, застосованого конкретно до експертної підсистеми для забезпечення децентралізації та цілісності експертних даних, що унеможливлює їх фальсифікацію після подання.

Крім того, переглянуто економічну модель експертної підсистеми з використанням реальних даних. Результати цього дослідження дозволяють створювати безпечні децентралізовані додатки на децентралізованих платформах даних з акцентом на зручність і простоту використання, а також демонструють інноваційне застосування в системі підтримки прийняття рішень для збору експертних знань.

Ключові слова: децентралізовані платформи даних, делеговані транзакції, системи підтримки прийняття рішень, експертні дані, блокчейн, Ethereum.

Tsyganok Vitaliy, doctor of technical science, professor, professor at the cybersecurity and application of information systems and technologies academic department, Institute of special communications and information security of National technical university of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”, Kyiv, Ukraine, ORCID 0000-0002-0821-4877, tsyganok@ipri.kiev.ua.

Savchenko Nikita, postgraduate student, Institute for information recording of National academy of sciences of Ukraine, Kyiv, Ukraine, ORCID 0000-0003-1107-0461, zitros.lab@gmail.com.

Tsyhanok Roman, student, National technical university of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”, Kyiv, Ukraine, ORCID 0009-0002-6156-3037, tsyganok2018@gmail.com.

Циганок Віталій Володимирович, доктор технічних наук, професор, професор кафедри кібербезпеки і застосування інформаційних систем і технологій, Інститут спеціального зв’язку та захисту інформації Національного технічного університетуУкраїни “Київський політехнічний інститут імені Ігоря Сікорського”, Київ, Україна.

Савченко Микита Миколайовия, аспірант, Інститут проблем реєстрації інформації Національної академії наук України, Київ, Україна.

Циганок Роман Віталійович, студент, Національний технічний університет України “Київський політехнічний інститут імені Ігоря Сікорського”, Київ, Україна.