Beyond Self-Custody:
Delegated Authority in Tokenized Securities [*] [†]

Seoyoung Kim
Associate Professor of Finance
Santa Clara University

Sachin Muralidharan
Quantitative Engineer
DV01

September 2026

[For *Journal of Investment Management*]

"Case Studies" presents a case pertinent to contemporary issues and events in investment management. Insightful and provocative questions are posed at the end of each case to challenge the reader. Each case is an invitation to the critical thinking and pragmatic problem solving that are so fundamental to the practice of investment management.

[*] We thank M. Apu II for his invaluable support and contributions.
[†] The authors may be reached at: srkim@scu.edu, sachin.m.dharan@gmail.com; 500 El Camino Real; Santa Clara, CA 95053.

Traditional finance has long distinguished among asset ownership, custody, and delegated authority. An investor may remain the beneficial owner of a security registered in street name, entrust its safekeeping to a custodian, and separately authorize a broker, investment advisor, or other agent to transact on the investor's behalf. Custody, in this setting, does not itself confer discretion to invest, lend, or otherwise deploy the securities. Conversely, an authorized agent may exercise substantial authority over the securities without ever holding them in custody.

Tokenization initially appeared to simplify this structure by presenting self-custody as an alternative to intermediary dependence. Rather than entrusting tokenized securities – securities whose ownership and transaction records are maintained on a blockchain – to a centralized intermediary, investors could hold the private keys controlling their whitelisted on-chain accounts, appear directly on the issuer's records, and interact with on-chain trading venues without routing their transactions through a conventional broker or custodian. In blockchain parlance, the custodial/noncustodial distinction consequently came to serve as a broader proxy for the presence or absence of third-party dependence: an investor who held the private keys was presumed to retain meaningful control over the assets and avoid the trust relationships associated with traditional finance.

More sophisticated on-chain arrangements expose the limits of that inference. Investors may retain custody of their tokens while granting smart contracts substantial authority to transfer, exchange, lend, pool, or otherwise deploy them. In a simple transaction, a user initiates a specific action, authorizes its execution, and retains direct control over whether and when to proceed. Other arrangements rely on more persistent, generalized, or layered permissions that functionally resemble delegated authority, even when neither the protocol nor the investor describes them in agency terms.

The comparison to traditional agency relationships is functional rather than exact. In traditional finance, an investor might submit a specific order to a broker or authorize an investment advisor to exercise discretionary judgment in selecting securities and placing trades. On-chain, comparable powers may instead be distributed across token approvals, smart-contract permissions, protocol rules, upgrade mechanisms, governance processes, and the interfaces through which users interact with the on-chain protocol. A nominally disintermediated arrangement may therefore eliminate the third-party custodian without eliminating third-party authority over self-custodied assets. Instead, it may unbundle powers historically exercised within regulated intermediary relationships and reallocate them among software protocols, developers, governance bodies, interfaces, liquidity providers, and other participants—without any one actor necessarily performing the full regulated role associated with the combined economic function.

This distinction becomes especially consequential as conventional securities move onto public blockchains and begin trading through on-chain alternative trading systems (ATS's) and related market infrastructure – a development that may be accelerated by the SEC's recent "Innovation Exemption." The order grants "temporary, conditional exemptive relief to Tokenized Securities Venues," permitting them "to trade tokenized National Market System (NMS) stock using innovative permissioned automated market makers and liquidity pools."[1] A whitelisted investor may remain both the registered and beneficial owner of a self-custodied tokenized security while authorizing smart contracts or trading protocols to transfer, exchange, pool, or otherwise affect that security. The investor may have no conventional custodian and may not have expressly engaged a wealth manager, broker, or other readily identifiable intermediary apart from the

[1] See U.S. Securities and Exchange Commission, "SEC Issues 'Innovation Exemption' to Facilitate the Trading of Tokenized NMS Stock and Request for Comment," Press Release No. 2026-90, September 17, 2026, accessed at https://www.sec.gov/newsroom/press-releases/2026-90-sec-issues-innovation-exemption-facilitate-trading-tokenized-nms-stock-request-comment.

relevant protocol or venue. Nevertheless, economically significant powers may be delegated across a combination of code, interfaces, governance systems, developers, operators, and liquidity providers.

Such delegated authority does not necessarily make a protocol equivalent to a broker, investment advisor, custodian, or other regulated intermediary. An investment advisor, for example, ordinarily exercises judgment on behalf of a client and is subject to licensing, conduct, disclosure, fiduciary, and other agency-based obligations. A protocol may instead execute a narrower set of permissions mechanically according to code. Yet those permissions may still determine when and how assets move, the counterparties and pricing mechanisms to which they are exposed, the contracts with which they may interact, and the conditions under which the investor can recover, revoke, or redirect them.

Securities tokenization therefore creates a potential mismatch between the allocation of economic authority and the regulatory categories traditionally used to govern it. Powers once bundled within identifiable and regulated institutions may become disaggregated across actors and technological components, none of which individually appears to perform the entire regulated role. The investor remains self-custodial, but the surrounding arrangement may be neither wholly self-directed nor free from delegated authority and third-party dependence.

That mismatch warrants careful examination before tokenized securities and on-chain trading venues become widely integrated into mainstream securities markets. The relevant question is not simply whether an investor retains the private keys or whether a protocol takes custody. It is which actors or systems can cause securities to move, under what authority, subject to what limitations, and with what mechanisms for revocation, recovery, oversight, and accountability. Once securities begin moving broadly through on-chain alternative trading systems, these design choices will no longer be peripheral technical details. They will help determine how control, responsibility, and investor protection are allocated across the market structure itself.

These distinctions are visible in the design and implementation of ArCoin, the digital shares of the Arca U.S. Treasury Fund and a useful early example of a natively issued tokenized security. ArCoin's design drew upon ERC-1404,[2] a proposed but never finalized extension to the widely adopted ERC-20 standard[3] intended to support restricted securities transfers. ERC-1404 subjects the `transfer()` and `transferFrom()` functions to compliance checks – thereby restricting transfers to eligible or whitelisted accounts – but is silent regarding restrictions on the `approve()` function that governs who can exercise delegated transfer authority through `transferFrom()`. This omission exposes an important distinction: regulating the movement of a security is not the same as regulating the delegation of authority to move it.

ArCoin's original deployment using TokenSoft's published ERC-1404 implementation preserved that distinction.[4] A token holder could grant a spender transfer authority through the `approve()` function, even though any subsequent exercise of that authority through the `transferFrom()` function remained subject to the token's transfer restrictions. The granting of authority was therefore technically unrestricted at the interface level, while its exercise was gated at execution. That deployment has since been deprecated and replaced by an active deployment

---

[2] See Gierlach, Ron, James Poole, Mason Borda, and Lawson Baker, "ERC-1404: Simple Restricted Token Standard," Ethereum Improvement Proposals, July 27, 2018, accessed at https://github.com/ethereum/eips/issues/1404.

[3] See Vogelsteller, Fabian, and Vitalik Buterin, "ERC-20: Token Standard," Ethereum Improvement Proposals, November 19, 2015, accessed at https://eips.ethereum.org/EIPS/eip-20.

[4] ArCoin's original contract using TokenSoft's ERC-1404 implementation was deployed on the Ethereum blockchain at contract address `0x43C3983778eC88f0c383134E812C0c58A819aDD0`, accessed via its proxy contract deployed at `0x62a6738d887f47e297676fab05b902709b106c64`.

using Securitize's DS Protocol.[5] The active ArCoin deployment adds another layer of complexity, operating through an upgradeable proxy whose current implementation logic is unverified.[6] Simulated calls to the active proxy presently show that an `approve()` call from an arbitrary address succeeds while holder-initiated transfers between existing holder addresses revert with the contract's own error message `Tokens Locked`,[7] demonstrating how on-chain ownership may be separated from the immediate ability to exercise control on-chain through direct transfers.

The Arca U.S. Treasury Fund's governing documents separately provide that its transfer agent maintains the official shareholder record and, in the event of a conflict, will update the blockchain to conform to that record.[8] Authority consequently resides not only in the holder's private key, but also in the transfer restrictions, upgrade rights, and off-chain records surrounding the token. Other proposed security-token standards sought to allocate these powers differently.

[5] ArCoin's later contract using Securitize's DS Protocol was deployed on the Ethereum blockchain at contract address `0xEB9b9d0aF960E9bA579F624Cbee86856bb1d7e90`, accessed via its proxy contract deployed at `0x252739487c1fa66eaeae7ced41d6358ab2a6bca9`.

[6] A "verified" (as opposed to an "unverified") contract refers to a smart contract whose compiled source code has been (as opposed to having not been) verified to match the on-chain bytecode that governs the backend execution logic. Although ArCoin's active proxy contract is verified, its implementation contract is not. In contrast, ArCoin's original implementation contract was verified and published.

[7] We tested the active ArCoin deployment using `eth_call`, a standard Ethereum JSON-RPC method that executes the deployed contract code against the blockchain's current state (i.e., at block `26029684`) without signing or broadcasting transactions to the public memory pool (i.e., mempool). Because the call does not alter the blockchain's state, its `from` address may be specified without access to that account's private key. Using the active proxy address (`0x252739487c1fa66eaeae7ced41d6358ab2a6bca9`), we simulated a transfer from the largest publicly reported holder to another existing holder; the call reverted with the contract's own error message, `Tokens Locked`, whereas a simulated `approve()` call from an arbitrary address returned successfully. These results reflect the contract state at block `26029684` and calls against subsequent blocks may produce different results if the now-active implementation contract is deprecated and replaced. Appendix A provides a sample sequence of CLI commands. The complete commands, parameters, and containerized execution environment can be found in Muralidharan, Sachin, 2026, "ArCoin Reproducibility Tests," *Dex-Research*, GitHub repository, accessed at https://github.com/SachinDharan/Dex-Research/tree/main/joim/arcoin.

[8] "Although peer-to-peer transactions are viewable on Ethereum, record ownership of the Fund's shares is reflected on the records of the Transfer Agent, which is regulated by the SEC. The Transfer Agent will reconcile the blockchain transactions with the Fund's records on at least a daily basis, but the Transfer Agent's records will constitute the official shareholder records of the Fund and govern the record ownership of ArCoins in all circumstances." See Arca U.S. Treasury Fund Prospectus, May 1, 2026, accessed at https://www.arcalabs.com/hubfs/BG-Arca%20US%20Treasury%20Fund%205-1-26%20Pro-v3.pdf.

The ERC-1400 family expressly recognized multiple forms of authority, including operators and controllers with powers that may extend to forced transfers.[9]

A common thread is that these technological powers are assigned to blockchain addresses, that presumably belong to identified, licensed, or otherwise accountable legal actors. The interface itself and on-chain implementation can only specify what a controller is technically permitted to do but cannot establish who controls that address or what duties accompany the power. Part of ongoing oversight, then, must ensure that where the on-chain power exists, the corresponding off-chain legal accountability travel with it.

The recently finalized ERC-1450 represents the other end of the spectrum.[10] Rather than allowing investors to transfer securities subject to compliance restrictions, it disables holder-initiated transfers and approvals and, instead, places transfer authority with a registered transfer agent.[11] This approach resolves the fragmentation of authority by reconcentrating it in a recognizable regulated intermediary. The contrast among these standards is, thus, not simply one of greater or lesser decentralization. Each makes a different choice about who may authorize movement, who may execute it, and where responsibility for that authority resides.

---

[9] The ERC-1400 interface comprises a suite of proposed ERC standards, including the ERC-1410 interface, which defines holder-delegated operators, and the ERC-1644 interface, which defines issuer-appointed controllers who can execute transfers without holder authorization to reverse fraudulent transfers, respond to court orders, and move funds to new accounts if holders lose their private keys. See Dossa, Adam, Pablo Ruiz, Fabian Vogelsteller, and Stephane Gosselin, 2018, "ERC-1400: Security Token Standards," *Ethereum Improvement Proposals*, accessed at https://github.com/ethereum/EIPs/issues/1411; Dossa, Adam, Pablo Ruiz, Fabian Vogelsteller, and Stephane Gosselin, 2018, "ERC-1410: Partially Fungible Token Standard," *Ethereum Improvement Proposals*, accessed at https://github.com/ethereum/EIPs/issues/1410; and Dossa, Adam, Pablo Ruiz, Fabian Vogelsteller, and Stephane Gosselin, 2018, "ERC-1644: Controller Token Operation Standard," *Ethereum Improvement Proposals*, accessed at https://github.com/ethereum/EIPs/issues/1644.

[10] See Marks, Howard, Devender Gollapally, Joe Mathews, Jordan Jahja, John Shiple, and David Zhang, 2018, finalized July 2026. "ERC-1450: RTA-Controlled Security Token." *Ethereum Improvement Proposals*, accessed at https://eips.ethereum.org/EIPS/eip-1450.

[11] Specifically, the ERC-1450 standard requires `transfer()`, `approve()`, and `transferFrom()` to revert and `allowance()` to return zero. Instead, tokens can only be transferred through functions controlled by the registered transfer agent (RTA). Although token holders and RTA-registered brokers can submit transfer requests, they must await RTA approval and execution.

Even these interfaces do not reveal every economically meaningful source of control. A proxy administrator may replace the implementation governing the token, while the party controlling the relationship between an investor's identity and whitelisted address may determine whether that investor holds the private key. The relevant question, therefore, pertains not only to who retains ownership and custody of the on-chain asset but also to who can authorize, prevent, compel, or redefine the movement of the security, and what mechanisms exist for limitation, revocation, recovery, oversight, and accountability.

Questions

1. What should an improved security token interface require?
2. Does ERC-1450 solve the authority problem or merely recreate traditional intermediation on-chain? That is, if only a registered transfer agent can move securities, what meaningful function does blockchain provide beyond a public, auditable record?
3. What would make genuinely on-chain securities trading viable? For instance, what must change in identity/KYC infrastructure, broker and ATS registration, and investor demand for self-custody before tokenized securities progress from recorded ownership to actual on-chain trading and settlement?

References

Arca U.S. Treasury Fund. 2026. *Arca U.S. Treasury Fund Prospectus*. May 1. https://www.arcalabs.com/hubfs/BG-Arca%20US%20Treasury%20Fund%205-1-26%20Pro-v3.pdf.

Dossa, Adam, Pablo Ruiz, Fabian Vogelsteller, and Stephane Gosselin. 2018. "ERC-1644: Controller Token Operation Standard." *Ethereum Improvement Proposals*. https://github.com/ethereum/EIPs/issues/1644.

Dossa, Adam, Pablo Ruiz, Fabian Vogelsteller, and Stephane Gosselin. 2018. "ERC-1410: Partially Fungible Token Standard." *Ethereum Improvement Proposals*. https://github.com/ethereum/EIPs/issues/1410.

Dossa, Adam, Pablo Ruiz, Fabian Vogelsteller, and Stephane Gosselin. 2018. "ERC-1400: Security Token Standards." *Ethereum Improvement Proposals*. https://github.com/ethereum/EIPs/issues/1411.

Etherscan. "ArCoin Active Implementation Contract (unverified), 0xEB9b9d0aF960E9bA579F624Cbee86856bb1d7e90." Accessed at block 26029684. https://etherscan.io/address/0xEB9b9d0aF960E9bA579F624Cbee86856bb1d7e90#code.

Etherscan. "ArCoin Active Proxy Contract (verified), 0x252739487c1fa66eaeae7ced41d6358ab2a6bca9." Accessed at block 26029684. https://etherscan.io/address/0x252739487c1fa66eaeae7ced41d6358ab2a6bca9#code.

Etherscan. "ArCoin Original Implementation Contract (verified), 0x43C3983778eC88f0c383134E812C0c58A819aDD0." Accessed at block 26029684. https://etherscan.io/address/0x43C3983778eC88f0c383134E812C0c58A819aDD0#code.

Etherscan. "ArCoin Original Proxy Contract (verified), 0x62a6738d887f47e297676fab05b902709b106c64." Accessed at block 26029684. https://etherscan.io/address/0x62a6738d887f47e297676fab05b902709b106c64#code.

Gierlach, Ron, James Poole, Mason Borda, and Lawson Baker. 2018. "ERC-1404: Simple Restricted Token Standard." *Ethereum Improvement Proposals*, July 27. https://github.com/ethereum/EIPs/issues/1404.

Marks, Howard, Devender Gollapally, Joe Mathews, Jordan Jahja, John Shiple, and David Zhang. 2018, finalized July 2026. "ERC-1450: RTA-Controlled Security Token." *Ethereum Improvement Proposals*. Final. https://eips.ethereum.org/EIPS/eip-1450.

Muralidharan, Sachin. 2026. "ArCoin Reproducibility Tests." *Dex-Research*. GitHub repository. https://github.com/SachinDharan/Dex-Research/tree/main/joim/arcoin.

U.S. Securities and Exchange Commission. 2026. "SEC Issues 'Innovation Exemption' to Facilitate the Trading of Tokenized NMS Stock and Request for Comment." Press Release No. 2026-90, September 17. https://www.sec.gov/newsroom/press-releases/2026-90-sec-issues-innovation-exemption-facilitate-trading-tokenized-nms-stock-request-comment

Vogelsteller, Fabian, and Vitalik Buterin. 2015. "ERC-20: Token Standard." *Ethereum Improvement Proposals*, November 19. https://eips.ethereum.org/EIPS/eip-20.

Appendix A. Reproducing the ArCoin `eth_call` Results

The two tests described in footnote 7 can be reproduced by invoking the following commands from any terminal with curl installed. Each command sends an `eth_call` request to ArCoin's active proxy contract on Ethereum (`0x252739487c1fa66eaeae7ced41d6358ab2a6bca9`) at block `26029684` (`0x18d2e74` in hexadecimal format). The request executes the deployed contract code against the blockchain's state at that block. Nothing is signed or broadcast, and no private key is required.

The commands use Tenderly's public gateway (https://mainnet.gateway.tenderly.co), which requires no account or API key. Any correctly configured Ethereum archive node should return the same results.

Test 1: Transfer between existing holders.

```
curl -s https://mainnet.gateway.tenderly.co -H 'Content-Type:
application/json' -d
'{"jsonrpc":"2.0","id":1,"method":"eth_call","params":[{"from":"0x0962
e5e430c512005e1345af2b05d260e9347099","to":"0x252739487c1fa66eaeae7ced
41d6358ab2a6bca9","data":"0xa9059cbb000000000000000000000000b6ffc6f5f6
c98dc08348760994fe55522fe614010000000000000000000000000000000000000000
000000000000000000000001"},"0x18d2e74"]}'
```

This command simulates a `transfer()` of one unit from the largest holder (`0x0962e5e430c512005e1345af2b05d260e9347099`) to another existing holder (`0xb6ffc6f5f6c98dc08348760994fe55522fe61401`). The data field begins with `0xa9059cbb`, the function selector for `transfer(address,uint256)`, followed by the recipient address and the amount.

The resulting output is as follows (the error's encoded data field is abbreviated):

```
{"id":1,"jsonrpc":"2.0","error":{"code":3,"message":"execution
reverted: Tokens Locked","data":"0x08c379a0..."}}
```

Test 2: Approval from an arbitrary address.

```
curl -s https://mainnet.gateway.tenderly.co -H 'Content-Type:
application/json' -d
'{"jsonrpc":"2.0","id":2,"method":"eth_call","params":[{"from":"0x0000
00000000000000000000000000000000dEaD","to":"0x252739487c1fa66eaeae7ced
41d6358ab2a6bca9","data":"0x095ea7b30000000000000000000000001111111111
1111111111111111111111111111110000000000000000000000000000000000000000
000000000000000000000001"},"0x18d2e74"]}'
```

This command simulates an `approve()` of one unit, called from an address that has never held ArCoin (`0x000000000000000000000000000000000000dEaD`). The data field begins with

`0x095ea7b3`, the function selector for `approve(address,uint256)`, followed by an arbitrary spender address and the amount.

The resulting output is as follows (a return value of `1` means true, so the approval succeeds):

`{"id":2,"jsonrpc":"2.0","result":"0x0000000000000000000000000000000000000000000000000000000000000001"}`

The same tests are available as a Python script, a shell script, and a Docker container at https://github.com/SachinDharan/Dex-Research/tree/main/joim/arcoin.