ERC-4519 NFTs for the Rental of Smart Homes

Subjects: Engineering, Electrical & Electronic

Contributor: Javier Arcenegui , Rosario Arjona , Iluminada Baturone

The rental of houses is a common economic activity. However, there are many inconveniences that arise when renting a property. The lack of trust between the landlord and the tenant due to fraud or squatters makes it necessary to involve third parties to minimize risk. A blockchain (such as Ethereum) provides an ideal solution to act as a low-cost intermediary. In particular, the use of non-fungible tokens (NFTs) based on ERC-4519 is very interesting for smart home tokenization. The ERC-4519 is an Ethereum standard for describing NFTs tied to physical assets, allowing smart homes (assets) to be linked to NFTs so that the smart homes can interact with the blockchain and perform transactions, know their landlord (owner) and assigned tenant (user), whether they are authenticated or not, and know their operating mode (NFT state). The payments associated with the rental process are made using the NFT, eliminating the need for additional fungible tokens and simplifying the process.

blockchain smart home Internet of Things IoT smart contract real estate non-fungible token NFT

1. Blockchains (particularly Ethereum) for the Rental of Smart Homes

In the literature, there are a few proposals for renting smart homes based on blockchains. The work in [1] proposes a solution to convert traditional rental agreements into smart contracts. The functions implemented in the smart contract are contract activation and signatures, deposit payments, water/electricity meter recordings, rent payments, completion of the rent process, refund of the deposit amount, acceptance/rejection of the deposit amount refund, and retrieval of the remaining rent balance. Similar functions are considered in the proposal in [2]. The main goal of these two works is to make the rental process secure, traceable, and visible, without the need for a central authority or intermediaries. Other works, such as [3], maintain the need for a trusted authority, but only for the arbitration and tracing of malicious users and to improve the feasibility of the solution with a reputation mechanism. In these works, the house is a passive actor in the process.

The work in [4] focuses on the buying and selling of smart homes in smart cities. If an individual requests access to a property, the house, acting as an IoT, receives the request, reads the blockchain, and approves access if the individual is the owner, or denies it otherwise. The work in [5] provides a solution for renting smart homes that includes the registration of an IoT device included in the smart home using the manufacturer's blockchain address and the device information (e.g., the electronic product code). The IoT device (for example, an IP camera) represents the smart home. The solution is based on a smart contract that defines the functionalities of checking

the ownership information, IoT device authentication, ownership transfer, and tenancy transfer. The smart contract is created by the manufacturer of the IoT device.

All the solutions referred to above are based on smart contracts. However, none of them employ home tokenization. The work in [6] introduces the tokenization of homes as a way to increase the security and liquidity of the real estate market and reduce the administrative burden and costs involved in buying and selling properties. Every token holder has some percentage of ownership in the home. The authors employ fungible tokens based on the ERC-777 token standard, which improves the widely used ERC-20 token standard to offer token holders more control over their tokens [7]. In the proposal in [8], each smart home has a gateway that manages and controls access to IoT devices by external services (such as garbage collection) according to smart contracts. They use fungible tokens for the payments of services but do not use tokens to represent the home.

None of the above-mentioned works use non-fungible tokens (NFTs). The most widely used token standard to define NFTs in Ethereum is the ERC-721 [9]. It describes the basic attributes that an NFT should possess, including its identifier and owner (and who can manage the NFTs of an owner), and provides the basic functions to track and transfer the ownership of NFTs (which can represent digital or physical assets). The work in [10] proposes a system for exchanging or selling real estate assets as ERC-721 NFTs.

Table 1 summarizes the main features of the above-mentioned works on blockchains and smart homes.

Table 1. Summary of related works concerning blockchains and smart homes.

Refs.	Focus on Rental	Focus on Property Transference	IoT-Based Access Control	Avoidance of Intermediaries	Home Tokenization
[1]	Yes	No	No	Yes	No
[2]	Yes	No	No	Yes	No
[3]	Yes	No	No	No	No
[4]	No	Yes	Yes	Yes	No
[5]	Yes	No	Yes	Yes	No
[6]	No	Yes	No	Yes	Yes (Fungible)
[8]	No	No	Yes	Yes	No
[10]	No	Yes	No	Yes	Yes (Non-Fungible)

Concerning tokens in Ethereum, the ERC-1155 (Multi-Token Standard) [11] was proposed to allow for any combination of fungible tokens, non-fungible tokens, or other configurations (e.g., semi-fungible tokens). A

limitation of ERC-721 NFTs and the ERC-1155 is that they do not consider users. Among the ERCs that consider users, the ERC-4494 (Permit for ERC-721 NFTs) [12] and the ERC-5334 (ERC-721 User And Expires And Level Extension) [13] are in the “Draft” state, which is the initial state of any EIP, so they may suffer variations before entering the “Final” state if they reach such a state.

Among the ERCs dedicated to rental NFTs, only the ERC-4907 (Rental NFT, an extension of ERC-721) is currently in the “Final” state [14]. It adds the user role and an expiration time for the use of the NFT. The ERC-5187 (Extend ERC-1155 with rentable usage rights) [15] and the ERC-5501 (Rental and Delegation NFT—ERC-721 Extension) [16] are currently in the “Draft” state. The ERC-2615 (Non-Fungible Token with mortgage and rental functions) is currently in the “Stagnant” state, which means that its authors did not solve the flaws or incorporate the suggestions for its improvement [17].

Table 2 shows the features of the above-mentioned ERCs and the ERC-4519. The ERC-4519 (Non-Fungible Tokens Tied to Physical Assets) [18], currently in the “Final” state, is the only standard that defines how to tie a physical asset (in this case, the smart home) to the NFT smart contract using a blockchain address. Since this tie is performed through the blockchain address, the asset can interact with the NFT smart contract by signing messages and transactions. Another advantage of the ERC-4519 is that it considers operating modes and allows for the establishment of secure communication channels between the physical asset, its owner, and its user. The following sections describe the ERC-4519 and its use for renting smart homes.

Table 2. Comparison of ERCs for NFTs.

Refs.	ERC Name	NFT with Users	NFT Tied to Asset	NFT with States	NFT Enabling Secure Communication	Status
[9]	ERC-721	No	No	No	No	Final
[7]	ERC-777	No	No	No	No	Final
[11]	ERC-1155	No	No	No	No	Final
[12]	ERC-4494	Yes	No	No	No	Draft
[13]	ERC-5334	Yes	No	No	No	Draft
[14]	ERC-4907	Yes	No	No	No	Final
[15]	ERC-5187	Yes	No	No	No	Draft

Refs.	ERC Name	NFT with Users	NFT Tied to Asset	NFT with States	NFT Enabling Secure Communication	Status
[16]	ERC-5501	Yes	No	No	No	Draft
[17]	ERC-2615	Yes	No	No	No	Stagnant
[18]	ERC-4519 [18]	Yes	Yes	Yes	Yes [9]	Final

ets

tes of the

ERC-721 include *tokenId*, *owner*, and *approved*. The attributes of the ERC-4519 can be described as follows: *tokenId* is a numeric value that identifies the NFT; *owner* is a blockchain address that identifies the owner of the NFT; *asset* is a blockchain address that identifies the physical asset tied to the NFT; *user* is a blockchain address that identifies the user of the NFT; *approved* is a blockchain address that indicates who can transfer the NFT; *state* is a numeric value associated with the NFT states ("*waitingForOwner*", "*engagedWithOwner*", "*waitingForUser*", and "*engagedWithUser*") that indicates whether the *owner*/*user* and the *asset* are authenticated to each other; *hashK_OA* is the hash value of the shared key between the owner and the asset; *hashK_UA* is the hash value of the shared key between the user and the asset; *dataEngagement* is a numeric value containing temporary data for the authentication process; *timestamp* is a numeric value containing the last time the asset executed the smart contract; and *timeout* is a numeric value containing the maximum time set for two executions of the smart contract by the asset (after this time, the asset is considered out of service). **Figure 1** shows the flowchart of the NFT states, with the *asset* and *user* addresses defined.

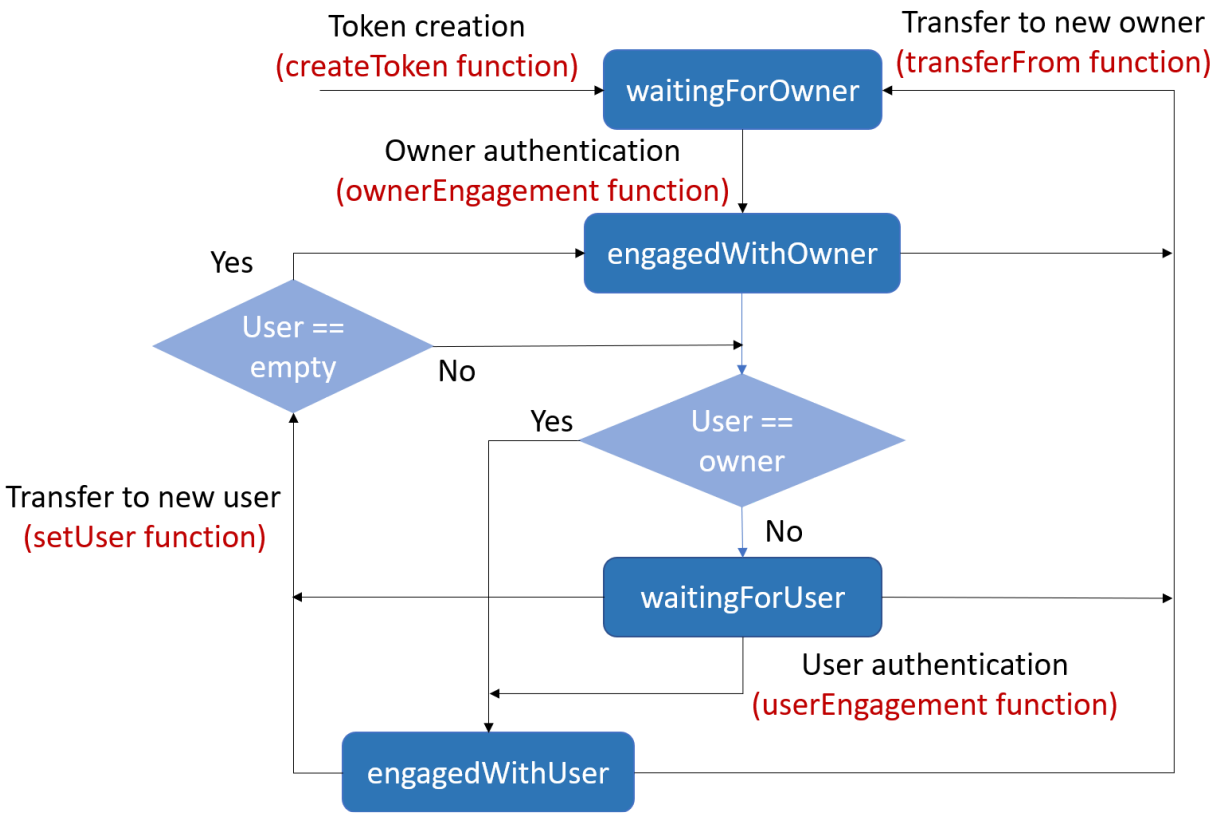

Figure 1. Flowchart of the ERC-4519 NFT states, with the *asset* and *user* addresses defined.

Regarding ERC-4519 events, the *UserAssigned* event is emitted when the NFT is assigned to a new *user*, the *UserEngaged* event is emitted when the *user* and the *asset* successfully complete the mutual authentication process, the *OwnerEngaged* event is emitted when the *owner* and the *asset* successfully complete the mutual authentication process, and the *TimeoutAlarm* event is emitted when the *timestamp* of the NFT is not updated within the *timeout*. None of these events are included in the ERC-721.

The ERC-4519 functions are as follows: *setUser* (which is executed by the *owner*) defines the new *user* of the NFT and changes its state to "*waitingForUser*"; *startOwnerEngagement* (which is executed by the *owner*) defines the initialization of the mutual authentication process between the *owner* and the *asset*; *ownerEngagement* (which is executed by the *asset*) completes the mutual authentication process between the *owner* and the *asset* if *hashK_OA* matches *hashK_A* (hash of the secret generated by the *asset* to share with the *owner*), changes the NFT state to "*engagedWithOwner*", and emits the *OwnerEngaged* event; *startUserEngagement* (which is executed by the *user*) defines the initialization of the mutual authentication process between the *user* and the *asset*; *userEngagement* (which is executed by the *asset*) completes the mutual authentication process between the *user* and the *asset* if *hashK_UA* matches *hashK_A* (hash of the secret generated by the *asset* to share with the *user*), changes the NFT state to "*engagedWithUser*", and emits the *UserEngaged* event; *checkTimeout* (which can be executed by everybody) checks whether the *timeout* has expired and emits the *TimeoutAlarm* event; *setTimeout* (which is executed by the *owner*) sets the value of the *timeout* attribute; *updateTimestamp* (which is executed by the *asset*) updates the *timestamp* attribute, thus avoiding the *timeout* alarm; *tokenFromBCA* (which can be executed by anyone) enables the retrieval of the attribute values of the *tokenId* from an address; *ownerOfFromBCA* (which can be executed by anyone) allows for the determination of the *owner* of the token from the address of the *asset* tied to the token; *userOf* (which can be executed by anyone) allows for the determination of the *user* of the token from the *tokenId* attribute; *userOfFromBCA* (which can be executed by anyone) allows for the determination of the *user* of the token from the *asset* attribute (address of the *asset* tied to the token); *userBalanceOf* (which can be executed by anyone) allows for the determination of the number of tokens assigned to a *user*; and *userBalanceOfAnOwner* (which can be executed by anyone) allows for the determination of the number of tokens of a particular *owner* assigned to a *user*. None of these functions are included in the ERC-721.

Depending on the application, some or all of the attributes, events, and functions described above may be used. For applications such as the rental of smart homes that require users (tenants) and a tie between the physical asset (smart home) and the NFT, all the functionalities are considered. Also, the application of renting smart homes requires additional attributes, events, and functions as described in the following.

3. Using ERC-4519 NFTs for the Rental of Smart Homes

The scheme of the proposed system for applying the ERC-4519 to the rental of smart homes is shown in **Figure 2**.

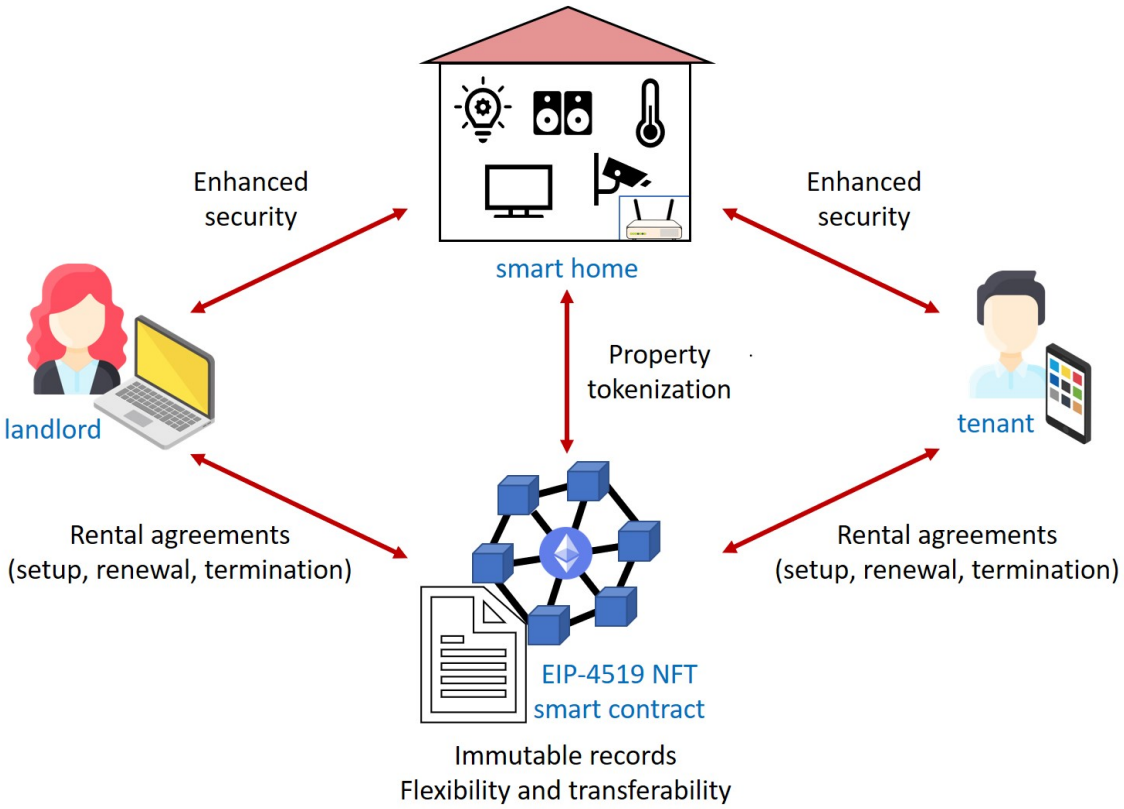

Figure 2. Scheme of the smart home rental using ERC-4519 NFTs.

Table 3 summarizes the ERC-4519 attributes considered for the rental housing application. The attributes on the left are typical of ERC-4519 NFTs, whereas those on the right are typical of smart homes represented by ERC-4519 NFTs.

Table 3. Attributes considered in smart homes represented by ERC-4519 NFTs.

Type	Attributes	Type	Attributes
uint256	*tokenId*	uint256	*deposit*
address	*owner*	uint256	*rentalPrice*
address	*user*	uint256	*waterMeter/waterPrice*
address	*asset*	uint256	*dieselMeter/dieselPrice*
enum States	*state*	uint256	*gasMeter/gasPrice*
uint256	*hashK_OA*	uint256	*electricityMeter/electricityPrice*
uint256	*hashK_UA*	uint256	*rentalTime*
uint256	*dataEngagement*	uint256	*homeIssues*
uint256	*timestamp*	struct	*Expenses_Meters*

Type	Attributes	Type	Attributes
uint256	*timeout*	struct	*Expenses_Prices*

measure
between
the end devices, both amongst themselves and with the cloud. It also manages data to/from the end devices and provides security to the smart home by mitigating risks. Since the IoT gateway is the central device of the smart home, it is considered the asset tied to the ERC-4519 NFT.

The process of renting a smart home requires a payment agreement for the temporary use of the property. Typically, the tenants have to pay an initial deposit to guarantee that they will pay any potential expenses and damages, a rental price (which may include regular charges incurred by the owner), and the water, diesel, gas, or electricity charges. To take this into account, the ERC-4519 should include the following additional attributes: *deposit* and *rentalPrice*, as well as *waterMeter*, *dieselMeter*, *gasMeter*, and *electricityMeter* to register, respectively, the water, diesel, gas, and electricity consumption. If some of these attributes are not needed, they can be set to 0. The temporary use is established by the *rentalTime* attribute. The price of each expenditure is determined by the *waterPrice*, *dieselPrice*, *gasPrice*, and *electricityPrice* attributes.

Finally, a smart home has the ability to check its status and determine whether the tenant is causing damage. These issues must be considered by the landlord to determine the cost through the *homeIssues* attribute. If necessary, the landlord will deduct the cost from the security deposit.

In the rental process, it is assumed that the smart home is equipped with a gateway and sensors and is owned by a landlord. Prior to the rental process, it is also assumed that an ERC-4519 NFT smart contract has been developed, with the *asset* attribute associated with the Ethereum address of the smart home gateway and the *owner* attribute associated with the Ethereum address of the landlord.

The landlord can modify the price of any expenses by executing the *setExpensesPrices* function, which sets the prices of water, diesel, gas, and electricity (0 if the service is not needed). Then, the *Expenses_Prices* attribute (a struct with all the prices) is modified. These prices can be determined by executing the function *getExpensesPrices*, which can be executed by anyone.

If the smart home gateway detects an issue during the rental period, it executes the *newHomeIssue* function. This function updates the *homeIssues* attribute to register the new issue and the *timestamp* attribute and sends the *HomeIssue* event to the landlord to notify that a new issue has been produced.

The functions provided in the smart contract are shown in **Table 4**. The functions depicted in the 7 upper rows are typical of ERC-4519 NFTs, whereas the others are typical of smart homes represented by ERC-4519 NFTs. These functions are used in the rental setup, rental renewal, and rental termination processes.

Table 4. Functions of the smart contract.

Function Name	Input Attributes
createToken	*asset, owner*
transferFrom	*tokenId, addressFrom, addressTo*
setUser	*tokenId, user*
startOwnerEngagement	*tokenId, hashK_OA, dataEngagement*
ownerEngagement	*hashK_OA*
startUserEngagement	*tokenId, hashK_UA, dataEngagement*
userEngagement	*hashK_UA*
setupRenting	*tokenId, user, rentalTime, rentalPrice, deposit*
sendMeterReadings	*waterMeter, electricityMeter, gasMeter, dieselMeter*
renewRenting	*tokenId, newRentalTime, newRentalPrice*
updateMeterReadings	*waterMeter, electricityMeter, gasMeter, dieselMeter*
payRenewal	*tokenId*
contractTermination	*waterMeter, electricityMeter, gasMeter, dieselMeter*
contractCancellation	*tokenId*
newHomeIssue	-
fixIssues	*tokenId, issueCost*
setExpensesPrices	*waterPrice, electricityPrice, gasPrice, dieselPrice*
getExpensesPrices	*tokenId*
endTenancy	*tokenId*
calculateExpenses	*tokenId, waterMeter, electricityMeter, gasMeter, dieselMeter*

Table 5 shows the transaction costs of the functions associated with the ERC-4519 NFT smart contract. The functions *endTenancy* and *calculateExpenses* are not included because these functions are executed through other functions. The gas price considered was 17 gwei (1 gwei = ETH 0.000000001) and the price of an Ether (ETH) was USD 1887.51. These values were obtained on 30 June 2023 at 12.35 CET. The function with the highest cost is the smart contract deployment. However, it is only performed once. Similarly, the function *createToken* is only executed once for each smart home gateway. When comparing the costs with those of a real estate agency, it can be seen that the costs are much lower, maintaining the reliability of the tenant's payment.

Table 5. Transaction costs of the functions of the smart contract.

Function	Transaction Cost (Gas)	Transaction Cost (ETH)	Transaction Cost (USD)
Deployment	5,173,251	0.0879453	166.00
createToken	211,963	0.0036034	6.80
startOwnerEngagement	76,687	0.0013037	2.46
ownerEngagement	58,500	0.0009945	1.88
setupRenting	144,608	0.0024583	4.64
sendMeterReadings	131,562	0.0022366	4.22
startUserEngagement	86,856	0.0014766	2.79
userEngagement	53,683	0.0009126	1.72
setExpensesPrices	51,129	0.0008692	1.64
renewRenting	99,888	0.0016981	3.21
updateMeterReadings	121,306	0.0020622	3.89
payRenewal	63,155	0.0010736	2.03
newHomeIssue	60,468	0.0010280	1.94
contractCancellation	50,109	0.0008519	1.61
contractTermination	81,279	0.0013817	2.61
fixIssues	72,534	0.0012331	2.33
setUser	51,734	0.0008795	1.66
transferFrom	66,300	0.0011271	2.13

Table 6 compares this research with others reported in the literature. It can be seen that this research is the only one that considers the smart home as a non-fungible token and allows it to participate actively in the blockchain. In [1][2][3], smart contracts are used to replace physical contracts and eliminate intermediaries. In [5], the smart home is recognized on the blockchain as an entity that does not directly interact with the blockchain and delineates the tenant and landlord off-chain. In researchers' work, the smart home is a token that performs on-chain authentication. Since the home is smart and has the ability to generate its own blockchain account, it can interact with the blockchain to authenticate its tenant and owner, and similarly, both the tenant and owner can authenticate the smart home. Concerning transaction costs, only the work in [1] provides the total cost of renting. This is shown in **Table 6**, expressed in Ethers. On the other hand, the total cost shown for this research considers the renting of a smart home already engaged with its owner (i.e., taking into account the transaction costs shown in **Table 5** for the functions *setupRenting*, *setUser*, *startUserEngagement*, *userEngagement*, *setExpensesPrices*,

Blockchain Cybersecur. Trust. Priv. 2020, 79, 33–50.

6. Gupta, A.; Rathod, J.; Patel, D.; Bothra, J.; Shanbhag, S.; Bhalerao, T. Tokenization of real estate using blockchain technology. In Proceedings of the Applied Cryptography and Network Security Workshops: ACNS 2020 Satellite Workshops, Rome, Italy, 19–22 October 2020; Proceedings 18. Springer International Publishing: Berlin/Heidelberg, Germany, 2020; pp. 77–90.
7. EIP-777: Token Standard. Available online: https://eips.ethereum.org/EIPS/eip-777 (accessed on 31 July 2023).
8. Al Omiri, H.H.; Al Husain, Z.; Rafeh, R. Integrating Blockchain and Internet of Things for Smart Homes. In Proceedings of the Computing, Communications and IoT Applications (ComComAp), Shenzhen, China, 26–28 November 2021; pp. 77–82.
9. EIP-721: Non-Fungible Token Standard. Available online: https://eips.ethereum.org/EIPS/eip-721 (accessed on 31 July 2023).

senderTenant, *Readings*, and *combination*) concerning execution times. Only the work in [3] provides the generation time of the key needed for the authentication process, which is performed off-chain between the tenant and the landlord. This is shown in **Table 6**, expressed in seconds. On the other hand, the generation time of the key shown for this research considers the time needed by the smart home to generate the public key that allows it to authenticate with the landlord and the tenant.

Table 6. Comparison of this research with state-of-the-art works.

Considerations	[1]	[2]	[3]	[5]	This Research
Smart home tokenization	No	No	No	No	Yes
On-chain authentication between home and tenant	No	No	No	No	Yes
Transaction costs per rental (ETH)	0.01095	-	-	-	0.01021
Generation time of key for authentication (s)	-	-	117.3	-	0.021

5. Conclusions

The ERC-4519 NFT is a good choice for tokenizing a home for rental purposes due to the following advantages:

- The NFT contains the Ethereum address of the smart home gateway as one of the attributes, and the gateway is the only one able to generate that address. This way, the home's authenticity can be checked as a way to prevent fraud.
- The NFT contains the Ethereum address of the tenant, as well as the landlord's address. This prevents squatters from being users of the home.
- The NFT allows for online authentication between the landlord and the smart home, as well as between the tenant and the smart home. Hence, the impersonation of tenants, landlords, and the homes themselves is avoided.
- Experimental results show that the transaction costs of the relevant functions of the smart contract are quite affordable.

10. Hati, M.K.; Agrawal, A.; Bhatia, R.; Bhatia, C.A.; Tiwari, K. T-PASS: A Blockchain-based NFT Enabled Property Management and Exchange System. In Proceedings of the International Conference on Information Networking (ICOIN), Bangkok, Thailand, 11–14 January 2023; pp. 140–145.
11. EIP-1155: Multi Token Standard. Available online: https://eips.ethereum.org/EIPS/eip-1155 (accessed on 31 July 2023).
12. EIP-4494: Permit for ERC-721 NFT. Available online: https://eips.ethereum.org/EIPS/eip-4494
13.
14.
15. EIP-5187: Extend EIP-1155 with Rentable Usage Rights. Available online: https://eips.ethereum.org/EIPS/eip-5187 (accessed on 31 July 2023).
16. EIP-5501: Rental&Delegation NFT-EIP-721 Extension. Available online: https://eips.ethereum.org/EIPS/eip-5501 (accessed on 31 July 2023).
17. EIP-2615: Non-Fungible Token with Mortgage and Rental Functions. Available online: https://eips.ethereum.org/EIPS/eip-2615 (accessed on 31 July 2023).
18. EIP-4519: Non-Fungible Tokens Tied to Physical Assets. Available online: https://eips.ethereum.org/EIPS/eip-4519 (accessed on 31 July 2023).