Token Spammers, Rug Pulls, and Sniper Bots: An Analysis of the Ecosystem of Tokens in Ethereum and in the Binance Smart Chain (BNB)

Federico Cernera
cernera@di.uniroma1.it
Sapienza University of Rome

Massimo La Morgia
lamorgia@di.uniroma1.it
Sapienza University of Rome

Alessandro Mei
mei@di.uniroma1.it
Sapienza University of Rome

Francesco Sassi
sassi@di.uniroma1.it
Sapienza University of Rome

Abstract

In this work, we perform a longitudinal analysis of the BNB Smart Chain and Ethereum blockchain from their inception to March 2022. We study the ecosystem of the tokens and liquidity pools, highlighting analogies and differences between the two blockchains. We discover that about 60% of tokens are active for less than one day. Moreover, we find that 1% of addresses create an anomalous number of tokens (between 20% and 25%). We discover that these tokens are used as disposable tokens to perform a particular type of rug pull, which we call *1-day rug pull*. We quantify the presence of this operation on both blockchains discovering its prevalence on the BNB Smart Chain. We estimate that 1-day rug pulls generated $240 million in profits. Finally, we present sniper bots, a new kind of trader bot involved in these activities, and we detect their presence and quantify their activity in the rug pull operations.

1 Introduction

The cryptocurrency market is loosely regulated [4, 39]. Even if policymakers are moving towards building a safer environment for cryptocurrency investors [56], it is a complex task, and needs time. Meanwhile, blockchain-related technologies evolve fast, and with the birth of the DeFi [67] investors begin to move from centralized exchanges (CEX) like Binance to decentralized exchanges (DEX). DEXes are distributed Applications (dApp) for trading that run on-chain powered by smart contracts. While regulating the standard cryptocurrency market is difficult, ruling the on-chain trading platform is even more challenging. Indeed, even if the web interface of a DEX can be shut down [3], its smart contracts are still reachable and working on the blockchain.

DEX and DeFi dApp were born in the Ethereum blockchain, but DeFi services rapidly pop up on all the blockchains that support smart contracts. Although Ethereum plays a leading role in the DeFi world, with over $68 billion locked in its smart contracts, the BNB smart chain or BSC (former Binance Smart Chain) proposes itself as a faster and cheaper alternative.

Uniswap and PancakeSwap are the two most popular DEXes on Ethereum and BSC. They rely on the Automated Market Maker (AMM) model to handle the trading system. At the basis of the AMM model, there is the concept of liquidity pools, a smart contract that handles two tokens (trading pair) that the user can swap. Unlike CEX, where the platform defines the trading pairs, users can create their pair on DEXes and let the other users use it. However, as we will see in the following, some users abuse this freedom to carry out a series of malicious operations.

In this work, we conduct a longitudinal investigation of tokens and liquidity pools in the Ethereum and BSC blockchains. We start by parsing over 3 billion transactions of both blockchains, finding more than 1.3 million tokens and 1 million liquidity pools (Sec. 4). Then, we reconstruct their lifetime—the time from their creation to their last transfer, discovering that approximately 60% of the tokens have a lifetime shorter than one day (Sec. 5). Hence, we define them as *1-day tokens*. A tiny fraction of addresses, just 1%, is responsible for creating more than 20% of the tokens (Sec. 6). Surprisingly, we also find that the tokens with a very short lifetime are actively traded on liquidity pools. Albeit this phenomenon is present on both blockchains, it is more widespread on BSC. Diving into this subset of tokens, we observe that a large fraction of liquidity pools used to trade the 1-day tokens show a malicious pattern that we call *1-day rug pull* (Sec. 7). We analyze all the liquidity pools looking for this pattern, and we find 272,349 potential rug pulls on BSC and 21,742 on Ethereum. We estimate the cost of the operation and the gains of the organizers, finding that they earned approximately $240 million with such activity (Sec. 7.2.1). Here, we see that the success rate of the 1-day rug pull is not very high (between 40% and 60%). However, given the simplicity and the very low cost of the operation, attackers can serially arrange the rug pulls and cover a series of unsuccessful operations with a single successful one. Finally, we study how this kind of operation evolved over time, discovering that the BSC has gradually

surpassed Ethereum in terms of the number of operations and gains. Moreover, we find that the operations are more prevalent during two specific events: the 2020 Defi Summer and the 2021 Altcoin season (Sec. 7.2.2).

Our key contributions are:

- **Analysis of BNB smart chain**: To the best of our knowledge, we are the first to study this young but well-established blockchain, performing a longitudinal analysis from its inception to March 2022. We study the tokens and the liquidity pools ecosystem, highlighting analogies and differences with Ethereum.
- **Short lifetime tokens and Token spammers**: We estimate the lifetime of the tokens on both blockchains. Discovering that about 60% of tokens last less than one day. Analyzing who creates the tokens, we observe that just 1% of addresses create an abnormal number of tokens (about 20-25% of tokens of the blockchains).
- **1-day rug pulls**: We investigate the presence of the rug pull pattern in 1-day tokens. We discover that on BSC, 81.2% of 1-day tokens listed on PancakeSwap have this pattern. We estimate the gains of the attackers, observing that even if the operation is very simple to arrange, given its cheap cost, it is profitable when performed serially.
- **The sniper bot 2.0**: We find the presence of sniper bots (Sec. 8), a particular kind of trader bot that observes the blockchain's mempool to buy newly listed tokens. To the best of our knowledge, we are the first to illustrate how this kind of trading bot works, detect their presence, and quantify their activity in the rug pull operations.

2 Ethereum and BNB Smart Chain

Ethereum [11] is a proof-of-work[1] blockchain. Its native coin is the Ether (ETH), the second most popular cryptocurrency after Bitcoin (BTC), with a market cap of more than 210 billion USD. A key feature of Ethereum is smart contracts, pieces of code that execute in a decentralized way on-chain. Through smart contracts, it is possible to create new digital assets like (fungible) tokens and NFTs (non-fungible tokens).

The tokens. Tokens are cryptocurrencies that can be exchanged or traded. They are created on top of the blockchain, and their mechanisms are defined using smart contracts. On Ethereum, the ERC-20 [26] standard defines the main properties of tokens. An ERC-20 compliant smart contract must implement a set of functions and events specified in the standard. These functions are reported in Table 5. Some of them are optional, in particular the *name()*, the *symbol()*, and the *decimal()* functions. On Ethereum, tokens and digital assets are held into accounts.

Ethereum accounts. There are two kinds of accounts on Ethereum: Externally owned accounts (EOA) and contract accounts. EOAs consist of a pair of public and private keys generated with the Elliptic Curve Digital Signature Algorithm (ECDSA) [35]. An account is represented by its public address, a 42-character hexadecimal string obtained concatenating "0x" to the last 20 bytes of the Keccak-256 [23] hash of the public key. Instead, a contract account is tied to a smart contract. It is represented with an address in the same format as an EOA. A contract account is generated when a smart contract is deployed to the Ethereum blockchain.

Transactions and fee. A transaction is an action that updates the whole Ethereum network. It can be used to move digital assets, deploy a smart contract, or invoke a smart contract. Executing a transaction has a cost, commonly called transaction fee. The fee is variable and depends on two main factors: The state of the network (if the network is heavily loaded, the fee is usually higher) and the complexity of the operation that the transaction triggers.

Smart contract deployment. Smart contracts are programs that run on the Ethereum blockchain. They are written in a high-level programming language (*e.g.,* Solidity [21]) and compiled into bytecode that runs on the Ethereum Virtual Machine (EVM) [25]. A smart contract can be deployed by sending a contract creation transaction from an EOA to the zero address[2]. The contract creation transaction contains the bytecode of the smart contract. A smart contract can also create new smart contracts. Since a smart contract can start a transaction only in response to a transaction that triggers it, an EOA must trigger the generation of a new smart contract.

Events and logs. To facilitate the tracking of internal states of smart contracts, Ethereum provides Events and an internal Logs register. Each time an action changes the internal state of a smart contract, it can fire an Event that notifies the change.

EVM and EVM compliance. Ethereum is a distributed state machine that changes its state at each new block accordingly to a predefined set of rules. The EVM [65] is the entity that computes these changes in states. Other than Ethereum, other blockchains rely on the EVM (*e.g.,* BNB Smart Chain [7], Avalanche [53], Fantom [27]), and they use one of the standard EVM or a custom one. These blockchains are called EVM-compliant. They run the same (or with minimal change) smart contract written for Ethereum, use the same convention for the address, and handle states the same way as Ethereum.

The BNB Smart Chain. The BNB Smart Chain [7] (previously Binance Smart Chain) or BSC is a blockchain that was born in 2020. Its consensus is based on the PoSA [9] (Proof of Stake and Authority). The coin of both chains is the BNB (Build and Build, previously Binance Coin)—the third coin by market cap with over 46 billion of capitalization. As Ether for Ethereum, the BNB coin fuels the transactions on the BNB chain. Because of EVM compatibility, it is possible to create tokens in BSC similarly to Ethereum. However, in

[1] Switched to proof-of-stake from 2022-09-15

[2] 0x0000000000000000000000000000000000000000

this case, tokens follow the BEP-20 standard instead of the ERC-20.

3 AMMs, Uniswap and its forks

Uniswap [1] is one of the most popular decentralized applications (dApp). According to DefiLlama [41], a popular DeFi statistics aggregator, Uniswap is the 5^{th} dApp by TVL (Total Value Locked, amount of money locked into smart contracts) with over 6 billion USD.

Uniswap is the first dApp to use the *AMM model* successfully. This model relies on a mathematical formula to fix the price of assets and on the concept of liquidity pools and providers. A *liquidity pool* is a smart contract that contains two or more cryptocurrencies that the user can swap for another. Instead, a liquidity provider is a user who invests in the liquidity pool, providing cryptocurrencies to the smart contract. When a liquidity provider injects liquidity into the liquidity pool, the smart contract mints LP-tokens and gives them to the liquidity provider. The LP-token represents the share of the liquidity pool owned by the investor. When the liquidity provider desires to get back his cryptocurrencies, he transfers the LP-tokens to the smart contract. The latter burns the LP-tokens and provides the cryptocurrencies back to the investor. In Uniswap V2, each pool consists of a pair of ERC-20 tokens. The liquidity pool is divided into two parts, each containing a single token, and both have an equivalent value. Let a pool consist of x token A and y token B. At each swap, the pool preserves $x * y$. When a user swaps a token A for token B (the user adds token A to the pool and takes token B from the pool), x increases by a and y decreases by b, where b is computed so that $x * y$ does not change. The rate a/b of the exchange depends on the ratio of x and y in the pool. Consequently, the swap operation changes the current exchange rate. The value of token A decreases while the value of token B increases, and the two parts maintain the same value.

Because of the success of this model, the popularity of Uniswap and its open-source smart contracts [62], more than 50 protocols were born on several blockchains by forking Uniswap smart contracts in the last years. Uniswap is on its third version, but all its forks belong to the second version since the third one is under a Business Source License. For this reason, in this work, we focus on Uniswap V2 and its forks. One of the most popular forks of Uniswap is PancakeSwap, which lives on BSC. It is the 1^{th} dApp by TVL on this blockchain with over 4 billion USD locked in its smart contracts.

4 The Datasets

For our investigation, we build two different datasets: The *Token Dataset*, which contains all the ERC-20 (resp. BEP-20) tokens created, and the *Liquidity Pool Dataset*, which contains data about liquidity pools. Each dataset has two versions, one with data from the Ethereum blockchain and the other from the BNB Smart Chain.

We consider the whole history of both blockchains from their inception to March 2022. For the Ethereum blockchain, we process all the blocks from block 0 (2015-07-30) to block 14340000 (2022-03-07). For the BSC blockchain from block 0 (2020-04-20) to block 15854000 (2022-03-07). Given the large amount of data and the need to parse the entire blockchains multiple times, for performance reasons and to avoid overloading public nodes (*e.g.,* nodes provided by Binance [6] and Infura [34]) or services (*e.g.,* BscScan or Etherscan), we host and run an Ethereum and a BNB Smart Chain node. Finally, to query the blockchains and process the data, we use the Web3 [46] and the Ethereum-etl [44] Python libraries. Web3 is a collection of libraries that allow the interaction with a local or remote EVM-compliant node. Ethereum-etl allows extracting information from EVM-compliant blockchains and exporting it into formats like CSV or JSON. The data collection phase was performed on an Ubuntu 20.04 machine with AMD EPYC 7301 (16-Core Processor, 2.80 GHz), 1 TB of RAM, and 4 TB SATA SSD with 560/530 MB/s read and write speed. Data processing took between 24 and 72 hours each time we parsed the entire blockchain, depending on the kind of data retrieved.

4.1 The Token dataset

4.1.1 Gathering smart contracts

As a first step to building the Token dataset, we collect all the contract creation transactions issued by EOAs. As mentioned in Sec. 2, EOAs can deploy a smart contract by sending a *contract creation transaction* to the zero address. We process all the transactions in the considered time frame in BNB Smart Chain (2.6 billion transactions) and in Ethereum (1.4 billion transactions). We collect 2,195,399 and 4,420,389 contract creation transactions respectively.

However, tokens can also be created by a smart contract itself. Indeed, it could be the case that an EOA calls a smart contract method, and its execution generates a new ERC-20 (or BEP-20) compliant smart contract. In this case, the token is created with a so-called *internal transaction*. Despite the name, internal transactions are not real transactions, but rather calls performed by smart contracts. These kinds of transactions are stored off-chain—they are not visible simply parsing the blockchain.

To track the tokens created by internal transactions, we can operate in two ways: The first way is to re-execute all the transactions in the blockchain in the EVM and trace all the calls. This process is extremely expensive [57] from a computational point of view. The alternative is to scan the Event log looking for events that emit a *Transfer event*. The second

way is much faster and we estimate that it loses only 12% of the total number of tokens created by internal transactions. Moreover, the missing tokens are never been used, traded or transferred, and are thus of little importance for our study (we discuss in detail the impact of this choice in Sec. 11). So, we parse all the logs of both blockchains, searching for smart contracts that emit a Transfer event compliant with the ERC-20 (resp. BEP-20) interface. Then, we use Etherscan [36] and BscScan [37] to retrieve the transactions that created these smart contracts and all the information.

At the end of these two steps, we have a collection of 3,087,274 and 4,534,599 smart contracts extracted from BSC and Ethereum, respectively. For each of them, we store the following information: The *address of the contract*, the *block number* in which the smart contract has been generated, the block in which the smart contract emits its last event, the EOA that deployed the smart contract or in the case of internal transactions the EOA address that triggers the first smart contract, the amount of *gas used*, the cost of the gas unit (*gas price*), the *bytecode* of the smart contract, and if the smart contract has been deployed by an EOA or through an internal transaction.

4.1.2 Token identification

Smart contracts are not only used to create tokens, as well as not all smart contracts that emit a Transfer event are tokens (*e.g.,* NFT contracts). Thus, we need to identify which of the retrieved smart contracts are ERC-20 (resp. BEP-20) compliant. Unfortunately, this is not a trivial task, and in the last years several works [13, 14, 22, 28, 63], attempted to face this problem with several approaches that we describe in Sec. 10. For our analysis, we follow the approach proposed by [14,63] that leverage the bytecode of smart contracts.

According to the Solidity specification [40], in the bytecode, smart contract's methods are identified by signatures that consist of the first 4 bytes of the *Kekkack-256* hash of the method name and parameters' type. Thus, to verify if a bytecode of a retrieved smart contract represents an ERC-20 (resp. BEP-20) compliant token, we verify if it contains at least all the signatures of the ERC-20 (resp. BEP-20) mandatory methods. Tab. 5 in the Appendix shows the signature of the mandatory and optional methods of the ERC-20 and BEP-20 interfaces.

Of the 4,534,599 smart contracts' bytecodes retrieved on the Ethereum blockchain, we find that 389,348 (8.5%) are ERC-20 tokens compliant, and 381,551 (98%) of them also implement the optional functions of the ERC-20 interface. Instead, on the BNB Smart Chain, we find that 1,887,484 out of 3,087,274 (61%) are BEP-20 compliant, and, as for Ethereum, almost all of them also implement the optional methods of the BEP-20 interface. Although we found more smart contracts on Ethereum than in BSC (4,534,599 vs. 3,087,274), there are many more compliant tokens in BSC (1,887,484) than in Ethereum (389,348). This discrepancy suggests that BSC may be a more interesting environment to study tokens and, possibly, their misuse.

Table 1: An overview of the Token dataset.

	Ethereum		BNB Smart Chain	
Contracts	Total	ERC-20	Total	BEP-20
External	4,420,389	293,688	2,195,399	1,021,427
Internal	114,210	95,660	891,875	866,057
Total	4,534,599	389,348	3,087,274	1,887,484
Total (w/o LP)	-	**323,863**	-	**1,078,016**

Table 2: An overview of the Liquidity pools dataset.

	Ethereum		BNB Smart Chain	
Events	Uniswap	Others	PancakeSwap	Others
PairC.	65,098	5,483	941,220	30,907
Mint	1,399,599	512,319	21,944,474	5,027,980
Burn	824,359	243,482	7,339,286	2,481,023
Swap	54M	27M	571M	179M

Lastly, we retrieve all the information about the identified tokens such as the name, the symbol, the number of decimals, and the total supply. To do so, we use the Ethereum-etl library and the Contract Application Binary Interface (ABI) [24]. The ABI is an interface between two program modules. It contains the specification for encoding/decoding methods and structures to interact with the machine code and interpret the results. Through the library, it is possible to instantiate smart contracts in an object-oriented manner and call its methods using an appropriate ABI. We instantiate the token contracts using an ABI that contains the specifications of ERC-20 (resp. BEP-20) methods and call the *name()*, *symbol()*, *decimals()*, *totalSupply()* methods.

At the end of the process, we have a dataset of ERC-20 (resp. BEP-20) tokens containing all the information about the smart contracts described in Sec. 4.1.1 and the related tokens. Table 1 shows the number of smart contracts on both blockchains.

4.2 Liquidity Pools dataset

To create the Liquidity Pool dataset, we consider Uniswap, its forks, and the other protocols that leverage its smart contracts.

Uniswap has three main smart contracts: *Factory*, *Pair*, and the *Router*. The Factory contract is responsible for creating the smart contract that handles the liquidity pool and the LP-tokens. The Pair contract keeps track of the balances of the tokens in the pool and implements the AMM logic explained in Sec. 2. The Router contract offers the entry point to interact with the liquidity pools. Thus, it is possible to swap tokens and add or remove cryptocurrencies from a liquidity

pool by interacting with the Router. Each of these contracts implements a set of Events that notify their status changes.

To build our datasets, we parse the Event log of the Ethereum and BSC blockchains. Following, we report the events we look for and a brief description:

- **PairCreated:** This event is fired by the Factory contract each time a new liquidity pool is created. We find 972,127 and 70,581 PairCreated events emitted on BSC and Ethereum, respectively. From the event, we can obtain the transaction hash, the block of the creation of the liquidity pool, the address that created the liquidity pool, the address of the liquidity pool, and the addresses of the two tokens (the pair of the liquidity pool), the gas used and the price paid per gas. Analyzing the address that fired the event, we find that almost all the liquidity pools of BSC are created in PancakeSwap (96.8%), and almost all the liquidity pools of Ethereum are created in Uniswap (92.2%).
- **Mint & Burn**: The Pair contract emits a Mint (or Burn) Event each time an LP-token is minted (or burned). This occurs whenever a liquidity provider adds (or removes) tokens into a liquidity pool. Analyzing these events, we obtain the transaction hash and the block of the Mint (Burn) Event, the address of the liquidity pool, the address that added (removed) the liquidity, the number of LP-tokens minted (burned), the gas used, and the price paid for the gas. We find 26,972,454 Mint events and 9,820,309 Burn events on BSC, and 1,911,918 Mint events and 1,067,841 Burn events on Ethereum.
- **Swap:** This event is fired by the Pair contract each time a user swaps tokens in a liquidity pool. From the event, we obtain all the information related to the swap: The transaction hash, the block in which the swap occurs, the address of the liquidity pool used, the address that performs the swap, the number of tokens swapped, the gas used and the gas price. We find 750,508,160 events on BSC and 82,447,051 events on Ethereum.

Moreover, we complete our dataset collecting for each smart contract the block number in which it emits the last event. Tab. 2 describes the final dataset.

Given that LP-tokens are ERC-20 (resp. BEP-20) compliant tokens, they are already present in our Tokens Dataset. However, our goal is to study standard tokens and liquidity pools separately. Thus, as the final step, we get rid of the information related to the LP-tokens from the Tokens Dataset. The last line on Tab. 1 reports the number of tokens after we get rid of the LP-tokens.

5 The Lifetime of tokens

Our data collection revealed a surprisingly high number of tokens and liquidity pools on Ethereum and BSC. Services like CoinGecko [16] or CoinmarketCap [18] list about 13,000 cryptocurrencies on 602 centralized and decentralized exchanges. Therefore, it is unclear what is the role of the large majority of tokens in the blockchain ecosystem.

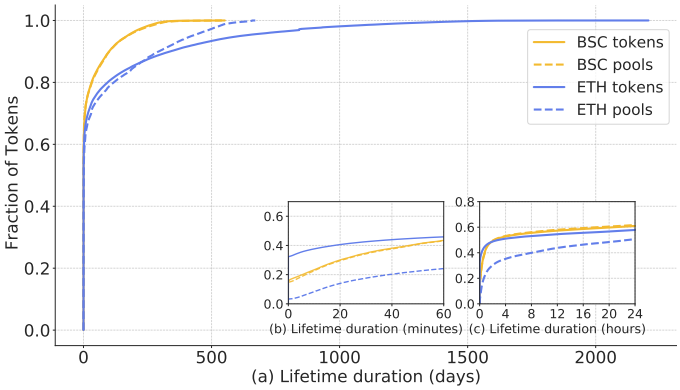

Figure 1: Lifetime of tokens and liquidity pools on BSC and Ethereum.

To obtain a first insight into the characteristics of tokens and liquidity pools, we introduce the concept of *lifetime*. We define the lifetime of a token in the following way: A token begins its lifetime at the block where its smart contract has been deployed, while it ends its lifetime in the last block where it emits any Event. Similarly, a liquidity pool begins its lifetime at the block where the PairCreated event is emitted, and it ends in the last block where the liquidity pool' smart contract emits any Event.

Fig. 1 shows the CDF of tokens' lifetime and liquidity pools' lifetime on Ethereum (blue lines) and BSC (yellow lines). Tokens and liquidity pools are shown with solid and dashed lines, respectively. The slope of the curves tells that the lifetime of the tokens in BSC is generally shorter than the lifetime of the tokens in Ethereum. Consider that BSC is a young blockchain, with slightly more than two years of activity (released on 2020-04-20), while Ethereum is more than seven years old (released on 2015-07-30). The longevity of Ethereum is also visible by the long tail of its tokens in the CDF. Nonetheless, it seems that Ethereum's tokens that tend to be more solid and long-lasting. This difference is smaller when we look at liquidity pools. Indeed, PancakeSwap, which handles about 97% of the liquidity pools in BSC, was born only four months after the release of Uniswap V2. From the CDF, we can also note a few additional interesting facts, particularly when we look at the first 24 hours of the life of tokens and liquidity pools.

A significant fraction of tokens is never active. Looking at the zoomed image in the center of Fig. 1 (b), it is possible to see that a significant fraction of tokens have a lifetime of length zero, meaning that the token is active only in one block, when it was created. This phenomenon is more common in Ethereum, with 104,836 out of 323,863 (32.4%) tokens that

Table 3: Summary of 1-day and 1-block tokens for BSC and Ethereum.

Lifetime	BSC	Ethereum
1-day	638,703 (59.2%)	187,378 (57.8%)
1-block	167,318 (15.5%)	104,836 (32.4%)
Total tokens	1,078,016	323,863

belong to this category, against 167,318 out of 1,078,016 (15.5%) in BSC. In the following, we refer to the tokens that last only one block as *1-block tokens*, while to the other tokens as *active tokens*. We find 910,698 and 136,485 active tokens on BSC and Ethereum, respectively. Table 3 succinctly reports on these statistics.

A large part of active tokens has an extremely short lifetime. Fig. 1 (b) shows that about 60% of the tokens in BSC and Ethereum have a lifetime shorter than one day. We refer to these tokens as *1-day tokens*. Considering only active tokens, we find that 471,385 (51.7%) of all the active BSC tokens and 82,542 (37.7%) of all the Ethereum active tokens are 1-day tokens. Looking at the data at a higher granularity (Fig. 1 (b)), we can note that the death ratio of BSC tokens is surprisingly high. Proportionally, BSC has approximately half of the 1-block tokens of Ethereum, about the same proportion of dead tokens after 60 minutes, and a significantly larger proportion of dead tokens after the first 4 hours. As we can see in Fig. 1 (c), the first four hours of token life are also crucial in Ethereum.

Almost all the BSC tokens with short lifetimes have a liquidity pool. Here, we find one of the main differences between BSC and Ethereum. 468,556 out of 471,385 (94.8%) active tokens with a lifetime shorter than one day in BSC have a liquidity pool. In Ethereum, only 33.1% (27,346). It seems that on BSC the liquidity pool is the main reason for creating a token.

6 Token spammers

In this section, we change perspective and explore who creates tokens. Retrieving the list of creator addresses from our token dataset, we find 144,795 and 464,095 different addresses that create at least one token, respectively, in Ethereum and BSC. Comparing these numbers with the total number of cumulative unique addresses on Ethereum (189,858,744) and BSC (140,522,222)[3], we see that they represent only a very small fraction of the addresses, the 0.07% in Ethereum and 0.33% in BSC. Fig. 2 shows the distribution of the number of tokens created by addresses in Ethereum and BSC. The first thing to notice is that the two distributions are extremely similar. The large majority of these addresses (70%) create only one token, as we can see in the zoomed image on the bottom right

[3]Data retrieved from Etherscan and BSCscan respectively

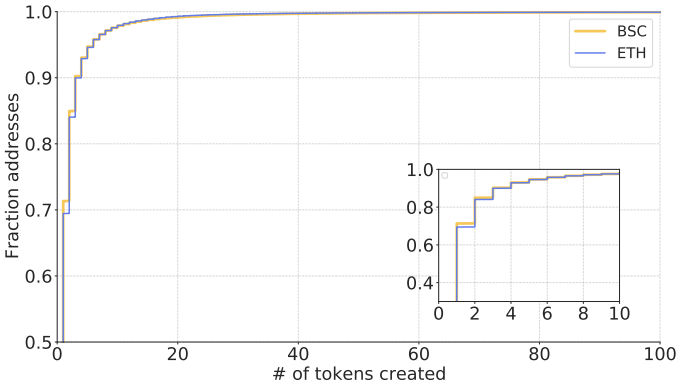

Figure 2: Distribution of the number of tokens created by the addresses that create at least one token in BSC and Ethereum. For the sake of visualization, the CDF is cut at 100 tokens.

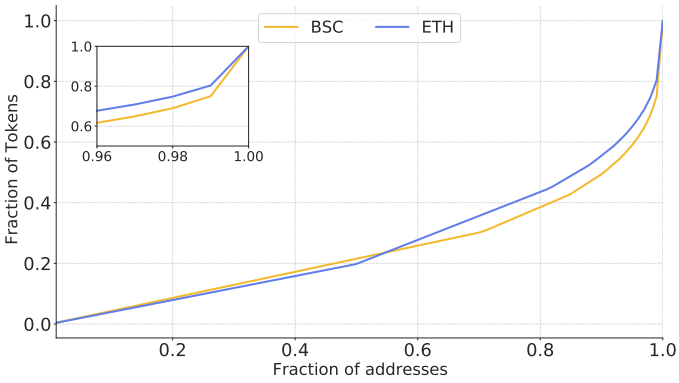

Figure 3: Fraction of addresses that create at least one token with respect to the fraction of tokens that they create.

corner of Fig. 2. 95% of addresses create five tokens or less, and just 1% of addresses create more than 18 tokens.

A small fraction of addresses creates a disproportionate amount of tokens. Fig. 3 shows the CDF of tokens created by fraction of addresses. From the figure, we can see that although 70% of addresses create just one token, the total amount of tokens created by these addresses account for only 30% of the tokens on the two blockchains. And more interestingly, we find that just 1% of the addresses creates 24.3% (262,023) of the tokens in BSC, and similarly, 1% of the addresses in Ethereum create 20.1% (67,869) of the tokens. These addresses create an average of 51 and 61 tokens in Ethereum and BSC, respectively. We will refer to these addresses as *token spammers*.

Token spammers are more prevalent in BSC. Although the distribution of the number of tokens created by addresses in Ethereum and BSC is almost identical (Fig. 3), the absolute numbers are different. Indeed, in terms of raw numbers, we find that BSC has almost four times more token spammers than Ethereum (4,231 vs. 1,329), and the spammers of BSC create almost four times more tokens in BSC than in Ethereum

(262,023 vs. 67,838).

Token spammers create tokens mainly with contract creation transactions. As mentioned in Section 2, tokens can be created in two ways: By sending a contract creation transaction or by sending a transaction to a smart contract that generates the token. We find that 94.8% of the tokens on BSC and 82.3% of the tokens on Ethereum are created directly by sending a contract creation transaction.

Token spammers create short lifetime tokens. As we have seen, a significant amount of tokens have a lifetime shorter than one day. Investigating the relationship between token spammers and 1-day tokens, we discover that most of the tokens created by the spammers have a lifetime shorter than one day. The spammers created 170,768 1-day tokens out of 262,023 (65.1%) and 40,552 1-day tokens out of 67,869 (59.8%), respectively, in BSC and Ethereum.

7 The Anatomy of a Rug Pull

The top token spammer creates 17,936 tokens in the timeframe of our analysis. If we look at the name of these tokens, we find that almost all of them have the same name (the tokens have only 76 unique names), with the most used being 'Pornhub' with 605 occurrences. The median lifetime of these tokens is extremely small: 45 mins. Lastly, almost all of the tokens (99.7%) created by this address have a liquidity pool. We study the liquidity pools of these tokens and find out that they are used to perform an operation commonly known as rug pull [43, 48]. In the following, we report a detailed example of a rug pull operation carried out by this address.

We focus on *OnlyFans*[4], a token created by the top token spammer on block 8090747 (2021-06-07 01:40:34 PM UTC) by issuing a contract creation transaction. On block 8090751 (2021-06-07 01:40:46 PM UTC), after 4 blocks from its creation, the token spammer creates a liquidity pool that contains the pair (OnlyFans, WrappedBNB) and adds a liquidity of 20 Wrapped BNB (almost $7,180 at the moment of the operation) and 44 trillion of OnlyFans tokens.

After just 6 seconds, on block 8090753, an address swaps 4 million OnlyFans for 0.002 Wrapped BNB ($0.718). This operation is followed by 11 other swaps—performed by 11 different addresses—for a total buy of $5.1740396 * 10^{12}$ OnlyFans for 2.67 Wrapped BNB ($958). After 2 hours from the creation of the token, at block 8093101 (2021-06-07 03:38:55 PM UTC), the token spammer removes all the liquidity from the liquidity pool, leaving it drained. Since the 12 addresses added Wrapped BNB into the pool by buying OnlyFans, the token spammer collects 22.67 Wrapped BNB and has a profit of 2.67 Wrapped BNB ($958).

We can formalize these operations in the following way:

1. Eve creates a new ERC-20 token τ.
2. Eve creates a new liquidity pool with pair (τ, B), where B is a valuable token, e.g. Wrapped BNB.
3. Eve adds liquidity to the liquidity pool. The reserves of the pool are now $(reserve_\tau, reserve_B)$.
4. At this point, Eve is the only one that owns token τ. Investors can buy token τ by swapping their tokens with token τ in the liquidity pool.
5. Suppose that Bob buys a few τ swapping it with B. The new reserves of the liquidity pool are $(reserve_{tau} - \delta_{tau}, reserve_B + \delta_B)$
6. Lastly, Eve removes all the liquidity from the liquidity pool. The net gain of the operations is δ_B minus the gas fees to execute the transactions.

An improved version of the operation. The rug pull described above is the simple version of the operation. However, to attract more investors, the attacker can manipulate some statistics of the liquidity pool. A well-known market manipulation that the attacker can use is *wash-trading* [12]. In this case, the creator of the pool tries to create the impression that the liquidity pool is active, faking the trading volume by repeatedly buying and selling tokens. Similarly, another way that attackers have to drum up the attention of investors is to inflate the price by buying the 1-day token gradually.
Finally, the attacker can also hedge his gains—eliminating the risk of an unrealized profit while the liquidity pool is still active. The attacker can maintain a reserve of tokens and, when investors start to buy the 1-day token, gradually sell the owned token, starting to take profit from the operation.

Clearly, rug pull operations can harm investors. However, we cannot consider it a "fraud" because the phenomenon is currently not regulated. In Appendix B we discuss this subject in depth.

7.1 Looking for 1-day Rug Pulls

We leverage our datasets to identify rug pulls systematically. Since we saw a considerable number of 1-day tokens and most of them are created serially, we narrow our investigation to the 332,265 in BSC and 25,180 in Ethereum 1-day tokens with a liquidity pool. Given the duration of these operations, we will refer to them as *1-day rug pulls*. We analyze all the Events emitted by the liquidity pools, looking for all the pools that emitted only one Mint and one Burn event in which the address that performs the transaction burns at least 99% of the minted LP-tokens (we don't use 100% since a small fraction of tokens might be stuck in the wallet due to rounding).

7.1.1 Estimating the gains of the operations

The simple operation, where the attacker does not swap in his liquidity pool, can be carried out by performing just four

[4] 0xe8b6f08841d668605343A63144D76ff2dE9A1199

transactions: A transaction that creates the token, one that creates the liquidity pool, one to add the liquidity, and finally, the last transaction to remove the liquidity. These transactions can be performed individually, or they can be aggregated by leveraging a smart contract. Of course, we consider both cases when computing the fees. If the attacker performs swaps on the liquidity pool, we also consider the transaction fees paid for each swap.

To perform our estimation we use the following formula:

$$base\_gain = \delta_B - fees \quad (1)$$

$$net\_gain = base\_gain - T_{in} + T_{out} - fees_{swap} \quad (2)$$

The formula can be split into two components. The first part computes the gain in the case of the simple operation.The second formula takes into account the improved version of the operation, where the creator of the liquidity pool manipulates it by performing swaps operations. In this case, we remove from the gain T_{in}, that is the amount of tokens that the manipulator artificially adds to the liquidity. We also add to the gain T_{out}, the quantity of tokens that the manipulator removes from the liquidity pool before the final removal of the liquidity (T_{out}). Finally, we remove from the gain the fees used to perform the swap operations (fee_{swap}).

7.2 Results

After processing our data, we discover that an incredibly high number of liquidity pools are actually rug pulls. In BSC, 272,349 out of 332,265 (81.2%) of the considered liquidity pools have a rug pull pattern, while 21,742 out of 25,180 (86.3%) in Ethereum. This result shows that attackers use most of the 1-day tokens as disposable to carry out rug pulls.

These operations are arranged by 116,516 different addresses in BSC and 16,539 different addresses in Ethereum. As we can expect from the previous analyses, most of the token spammers that operate in BSC are linked to this kind of activity. Indeed, in BSC, 2,112 out of 4,231 (50%) token spammers performed at least one rug pull. Instead, in Ethereum, there are only 45 token spammers (0.3%) that have been involved in this activity. We find 115 addresses that perform more than 100 rug pulls in BSC, accounting for 19.1% of the operations, with the most active performing 16,102 operations. Instead, in Ethereum, we find only one address performing more than 100. Interestingly, combining the information in the BSC and Ethereum dataset, we find a token spammer that operated on both blockchains with the same address [5]. He performs five rug pulls on Ethereum and three on BSC.

Looking at the liquidity pools, we find that BNB (97.8% of the cases) is the token paired the most with the 1-day token. It is followed by USDT (0.67%) and BUSD (0.15%), two stablecoins pegged to the USD. Instead, Wrapped Ether is paired with all the 1-day tokens in almost all the liquidity pools with a rug pull in Ethereum. As the next step, we want to estimate the number of users that fall prey to such activities. To do so, we exclude the addresses that swap into liquidity pools they have created themselves from this analysis. We collect 251,250 different addresses in BSC and 57,552 in Ethereum that interact with at least one liquidity pool with a rug pull pattern. These addresses performed 2,903,022 swaps on the considered liquidity pools in BSC and 317,257 in Ethereum.

We divide the swaps into buy (1-day token) and sell operations. As we can expect, given the anatomy of the 1-day rug pull, we find that most of the operations are buy operations. More in detail, in BSC 2,286,056 (78.7%) are buy operations and 616,966 (21.3%) sell operations. In Ethereum, we find a very similar pattern, with 254,061 (80.1%) buy operations and 63,196 (19.9%) sell operations.

As final metric, we compute the average value of the swaps performed by the users. The average amount of swaps is almost identical for buy and sell operations on both the blockchains, with 0.01 BNB for BSC and 0.19 ETH for Ethereum's liquidity pools. Interestingly, we notice a considerable difference in the average swap amount between the two blockchains. Indeed, the average swap is approximately $3 on BSC and $360 on Ethereum.

7.2.1 The gains

Before computing the gains of the attackers, we calculate the average price an attacker has to invest to arrange the operation. If the attacker does not perform any swap into the liquidity pool, the cost of the operation is on average 0.03 BNB in the case of BSC and 0.2 ETH for the Ethereum blockchain. Thus, the investment needed to perform such operations is low, even if it could vary substantially when the blockchains are overloaded. For instance, we found some rug pulls that reached the cost of 1.1 BNB or even 3.3 ETH. The base cost to arrange the operation is interesting because it represents a bound to the loss the attackers have to afford for each operation.

We leverage our datasets to compute the gain of the operation using the formula 1 described in Sec. 7.1.1. We describe the 266,340 operations on BSC and the 21,594 on Ethereum in terms of successful and unsuccessful operations based on the operation's net gain. In particular, we consider an operation successful if the net gain is strictly positive.

Successful operations. Among the liquidity pools with a rug pull pattern, there are 104,404 (39.1%) operations in BSC and 13,368 (61.9%) in Ethereum closed with a profit for the attacker. A possible reason for the higher success rate of the rug pull on Ethereum could be that, as we saw, on average, users tend to invest more money. Indeed, on average, attracting only one investor is enough to cover the operation's cost. To investigate what can affect the gains, we combine information on gains with those of the manipulations. When the creator of

[5] 0x87605612492c74bA0037fFaef676c0f3f6958918

the liquidity pool does not perform any kind of manipulation, the net gain is,on average 0.11 BNB in BSC and 1.34 ETH in Ethereum. Operations carried out on liquidity pools that suffer wash-trading activity have an average gain of 0.25 BNB in BSC and 12 ETH in Ethereum, which is considerably higher than the previous case. Instead, we notice a negligible increase in gains in the case of pump operations with respect to the gains obtained by the liquidity pools without manipulation. Moreover, we find that both kinds of manipulation have no impact on the success rate. This show that operations that have wash trading are generally more profitable. However, the attacker has to perform several swaps, increasing its cost and loss in case of an unsuccessful operation.

Unsuccessful operations. There are 161,936 (60.9%) liquidity pools in BSC and 8,226 (38.1%) in Ethereum, for which the attacker does not cover the transaction fees with the operations. For the 14% (21,122) of these liquidity pools of BSC and the 20% (1,506) of Ethereum, we notice that the operations were unsuccessful because nobody swapped into the liquidity pools. Considering the results we obtained, we can conjecture that the aim of the attackers is not to be successful every time but to arrange rug pulls serially and take profit in the long run. Indeed, the loss of an unsuccessful operation is minimal, and a streak of operations closed in loss can be covered with a single profitable operation.

Financial cost of 1-day rug pulls and comparison with other blockchain phenomena. In our study, we find that the number of 1-day rug pulls (21,594) and attackers (16,439) in Ethereum is significantly lower than in BSC (266,340 operations carried out by 117,110 rug pullers). Nonetheless, the total gain of Ethereum operations, around $150 million, is remarkably higher than the gains of BSC operations, that amount to $91 million. Moreover, the same trend holds when considering the volume of rug pull operations, which we define as the total value of BNB and ETH swapped. Here we find that Ethereum has a volume of $772.5 million against the $243.5 million of BSC. To gain insight into the magnitude of 1-day rug pull operations, we compare our metrics with popular blockchain shenanigans, like MEV and front-running. Tab. 4 in Appendix reports more relevant metrics collected from related works about operations carried out in Ethereum. As we can see, 1-day rug pull is the second type of operation by profit, generating slightly lower gains than Sandwich Attacks ($174.34 million in accordance with Qin et al. [50]). Particularly interesting is the number of addresses that perform the operations. Indeed, in Ethereum, the number of attackers that performed 1-day rug pulls is almost five times the number of the Sandwich Attackers (the fraud with the higher number of attackers in our comparison). We believe the operations are performed by a large number of addresses due to their ease of execution. The reported numbers highlight that 1-day rug pulls is a significant phenomenon in the DeFi ecosystems that involve hundreds of thousands of malicious actors and move more than 1 billion USD.

7.2.2 A longitudinal view

Fig. 4 provides a longitudinal view of the daily number of rug pull operations (Fig. 4 a), the liquidity added (Fig. 4 b) and the gains (Fig. 4 c). Analyzing the trends of the chart, we identify three different phases, divided by the black dashed lines in the figure. In the first phase, we find the first spike of 1-day rug pulls in Ethereum. In the second phase, rug pulls start to increase in the BSC. However, the Ethereum gains are generally higher for the same invested liquidity. Finally, in the third phase, we see that BSC surpasses Ethereum in terms of liquidity added, number of operations, and gains. In the following, we describe in detail the three phases:

Phase 1: DeFi Summer. The first phase took place approximately from June 2020 to March 2021. At the beginning of this phase, we see an increase in the daily number of rug pull operations in Ethereum, with a peak of 179 daily operations in October 2020. Then, the number of operations steadily decrease until March 2021. We believe that the increase in the number of operations was bootstrapped by a phenomenon known in the crypto-community as DeFi Summer 2020 [42]. During this period, DeFi became extremely popular, and, as a result, the market capitalization and prices of several tokens soared [51]. This interest in DeFi attracted new users looking for investment opportunities, which may have triggered the increase in rug pull operations. Fig. 4 (b) shows that there is a significant amount of liquidity invested in these operations, on average $37,941 (44 ETH), with an average gain of $5,969 (5.65 ETH) (Fig. 4). Note that this phase involves only the Ethereum blockchain because the BSC was released in September 2020 and was not very popular yet.

Phase 2: Altcoin season. Fig. 4 (a) shows a second spike in the number of rug pulls from March 2021 to September 2021. In this case, the spike involves both Ethereum and the BSC, which reach a maximum peak of 195 and 2,309 daily operations. It is interesting to notice that the number of operations over time follows the same trend for Ethereum and BSC. For this reason, we believe an exogenous event caused this spike. Analyzing the events of that period, we believe this rise in the number of operations may be a so-called *Altcoin Season*. An Altcoin Season is a period in which Altcoins[6] perform better than Bitcoin, significantly increasing their value. Previous study [38] shows that an Alt Season is marked by a drop of an indicator called *Bitcoin dominance*. This indicator measures the ratio between the market capitalization of Bitcoin to the total market capitalization of the entire cryptocurrency market. According to Coinmarketcap [18], in this period, the Bitcoin dominance decreased from 69% of January 2021 to 39% in May 2021. This market phase is frequently characterized by "Fear of missing out" (FOMO) [5], which makes investors more inclined to buy riskier tokens. For this reason, we believe investors have flocked to AMM markets to buy tokens,

[6]Altcoins [32] is a combination of the two words "alternative" and "coin". The term is used to indicate all cryptocurrencies except Bitcoin.

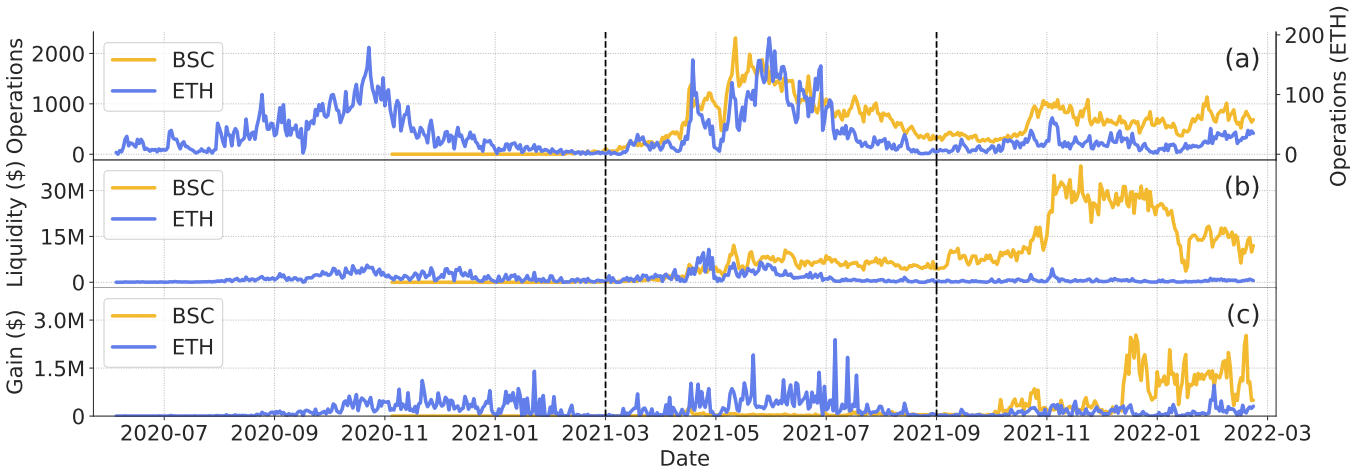

Figure 4: The figure shows the number of rug pull operations (a), the initial liquidity added to each pool (b), and the gain for each operation over time. All the metrics are aggregated daily. The dashed vertical lines divide the three phases we identify.

and rug pull operations skyrocketed. Fig. 4 (b) shows that the liquidity invested in these operations is higher for Ethereum, with an average of $39,625 (50 ETH) against the $5,624 (15 BNB) of BSC. Operations in Ethereum are also way more profitable, with an average gain of $5,836 (6.3 ETH) against the $48.4 (0.12 BNB) of BSC operations.

Phase 3: The overtaking of the BSC. The last phase goes from October 2021 to March 2022. In this phase, we find an interesting twist, as BSC surpasses Ethereum in terms of liquidity added and gains of rug pull operations. Indeed, in this phase, rug pulls in BSC have significantly more liquidity invested than in the past (56.9 BNB on average vs. 15.3 BNB of the previous phase) and higher gains (2.26 BNB on average vs. 0.12 BNB of the previous phase). For this reason, we can see in Fig. 4 that the total daily invested liquidity and gains in BSC are significantly higher than Ethereum and reached more than one million USD. In Sec. 11, we explore some possible reasons for this increase.

7.2.3 Tokens' names

To further deepen our analysis of rug pulls, we focus on the names used in the operations. Analyzing the rug pulls, we notice several tokens with the same name in BSC and Ethereum. We find that of the 272,349 tokens involved in the operations in BSC and 21,742 in Ethereum there are only 157,864 (57.9%) and 18,801 (86.4%) unique names. Thus, we attempt to cluster the 1-day tokens into categories and enumerate them. Table 6 in the Appendix shows the most used names and the number of occurrences for each of them.

As a first category, we explore clones—tokens with the same name as an existing (and more popular) cryptocurrency. To systematically search for these cases, we use as an authoritative source the CoinGecko APIs [16]. Leveraging them, we retrieve the names and the addresses of all tokens created and verified with the indexer service on the BSC and Ethereum. At the end of the process, we build a list of 5,325 tokens for BSC, and 5,172 tokens for Ethereum. We complement this list by adding popular variations for some tokens' names (*e.g.,* we also considered ADA as a possible name for the Cardano token). Using our list, we discover 22,002 cloned tokens in BSC and 1,781 in Ethereum. The most cloned tokens in BSC are Berryswap (370), Shiba Inu (191), and SafeMoon (158).

The second category we explore is the one of tokens that attempt to impersonate companies or websites. In this case, to obtain a list of possible target companies, we retrieve the name of the companies of the Standard and Poor's 500 (S&P 500) stock market index. Instead, for the websites, we extract from the Alexa ranking [7] the name of the top-ranked 200 websites. Using in conjunction these two lists, we find 4,638 tokens of this category in BSC and only 95 in Ethereum. The companies and websites that are present the most are Pornhub (1,023), Spacex(419), Onlyfans (398), Oracle (319), and Amazon (270).

We find several names that contain popular meme-related words like "Doge", "Inu" or "Shiba". This is not surprising, since meme tokens are very popular after the events that involved the "meme stocks" of GameStop (GME) and AMC Entertainment (AMC) in late 2020 [45]. Luckily, CoinMarketCap and CoinGecko offer a categorization of the tokens that also contain the "meme" category. We leverage these lists to extract the most frequent words and search for them into the tokens involved in rug pulls. We find a huge amount of tokens of this category: 54,229 in BSC and 4,835 in Ethereum.

As the last category of our investigation, we look for DeFi services (*e.g.,* Deriswap, Shibaswap, and Eco Finance). In this case, we simply search for tokens containing the "swap", "defi" and "finance" keywords. With this approach, we find for

[7] Data retrieved 2022-04-26

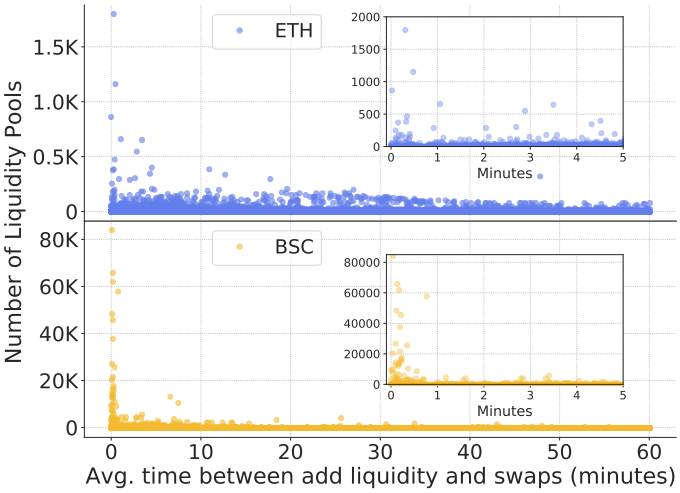

Figure 5: Scatter plot of the number of liquidity pools with a 1-day rug pull pattern where the address swapped and the average delay from the pool creation.

this category 25,524 tokens in BSC and 3,751 in Ethereum.

With our simple categorization, we covered the names of 39% of the 1-day tokens on the BSC and 48% on Ethereum. Even if we were not able to categorize all the tokens, we get some insights on how attackers pick the name to arrange their operations. In particular, we note a strong trend in choosing tokens' names related to the meme category and leveraging the name of popular cryptocurrencies, services, and companies.

8 Sniper Bots 2.0

We find that a large fraction of rug pulls are successful, even if they are zero-effort operations, without fake tokens or wash trading. Since these kinds of operations are very quick and simple, it is still unclear how they can be profitable. We analyze the operations carried out inside rug pulls more in-depth and discover that their success may be due to the activity of a particular class of trading bots, called Sniper Bots.

Sniper bots are automated bots that monitor time-bound activities and perform an action before or after anyone else. An example of sniper bot are "Scalping Bot", bots that monitor the availability of target products from a website and buy them as soon as they are available (*e.g.,* GPU Nvidia GPUs) [10].

With the birth of and the widespread adoption of AMMs, a new kind of sniper bot has been developed, which we define *Sniper Bots 2.0*. These kinds of sniper bots are programs that buy tokens on liquidity pools as soon as they are listed. To do so in the fastest way, sniper bots can leverage the mempool— the list of transactions not yet inserted in blockchain blocks. We find examples of these bots distributed for free on Github [20, 52, 60] and for a price at several other websites [2, 49]. Analyzing the code, we can infer how they work. As a first step, the sniper bot must search for newly listed tokens. The fastest implementation scans the mempool looking for transactions whose byte-code indicates that they are adding liquidity to a brand new liquidity pool. Another possibility is that the sniper bot waits for the token to be listed on services like BscScan or Etherscan. Then, the bot sends a swap transaction to buy the token, and if the gas price is properly adjusted, it is executed in the same block (but immediately after) of the transaction that adds the liquidity. Sniper bots typically execute only the buy operation. The user then can freely decide when to sell the token and make a profit. However, we also found some variants that automatically sell the token when the price reaches a pre-defined goal.

8.1 Identifying Sniper Bots

We conjecture that one of the reasons for the profitability of rug pulls operations are sniper bots that buy tokens from every liquidity pool indiscriminately. Thus, we can consider the liquidity pools involved in rug pulls as "honey pots" to detect sniper bots. To verify our intuition, we focus on addresses that swapped inside liquidity pools with a rug pull Fig. 5 shows the phenomenon: Every dot is an address, and its position indicates the number of different liquidity pools where the address swapped and the average delay from the pool creation. The figure shows a few addresses that swap in thousand of liquidity pools almost immediately after their creation. Since these addresses perform these operations serially and incredibly fast, we believe they must be sniper bots. We set up two conservative thresholds to identify evidence of addresses used by sniper bots.

For BSC, we consider all the addresses that swap on average with a delay smaller than five blocks (15 seconds) and that swap in at least 100 different liquidity pools. We flag 130 addresses as possible sniper bots. These addresses represent only 0.03% of all the addresses that swap inside liquidity pools involved in rug pulls. What is impressive is that they swap in 235,777 liquidity pools, representing 68.7% of all the liquidity pools with a rug pull. Moreover, these addresses also perform an impressive number of swaps: 2,691,173, that account for 24% of all the swaps performed in liquidity pools with a rug pull. We find that 31% of these swaps are performed in the same block where the liquidity is added for the first time in the liquidity pool. In these cases, we can confirm that the sniper bots scanned the mempool to swap in the same block where the liquidity is added. However, we also find sniper bots that perform the swap operations a few blocks after the liquidity is created.

We find sniper bots to be less present in Ethereum. Also, in this case, we pick two thresholds and consider all the addresses that swap on average with a distance lower than three blocks (45 seconds) and that swap in at least 10 liquidity pools. We find 64 possible sniper bots that swap in 30% of all the liquidity pools and perform a much smaller fraction of swaps with respect to BSC sniper bots (3.5% of the total). However,

interestingly, a higher percentage of swaps are performed in the same block where the liquidity is added in the liquidity pools (60%).

9 1-day Rug Pull Mitigation

Our study highlights that the 1-day rug pulls have some distinctive features. In the following, we propose some metrics that stem from the lessons learned from our analysis that may be useful to build a detection system.

- **Token lifetime:** This metric measures the time that elapses since the creation of the token. Indeed, we find that 1-day rug pull operations are performed in a very short timeframe (§7.1).
- **Distribution of the liquidity:** This metric tracks the distribution of the LP-tokens. In 1-day rug pulls, the liquidity pool creator owns all the liquidity (§7). Thus, it should be considered extremely risky when a single address owns most of the liquidity.
- **Address rug pull records:** This metric tracks addresses that performed a rug pull operation to add them to a list of potential malicious addresses. Indeed, we find that some addresses perform rug pulls multiple times. (§6).
- **Deceptive token name:** This metric measures the similarity between the name of tokens contained in the liquidity pools and popular existing tokens or companies. We find that attackers often deceive investors by exploiting the name of the token. (§7.2.3).

An attacker aware of these metrics can try to evade the detection by putting more effort into carrying out the operations (*e.g.,* using different addresses or creating the token in advance). Nonetheless, a distinctive characteristic of 1-day rug pulls is that they are easy to execute and require low effort by the attacker. Thus, we believe the proposed metrics could be sufficient to discourage this operation. Moreover, new metrics and more sophisticated techniques can be developed to identify attackers trying to circumvent the detection. For example, it is possible to follow the money flow between addresses associating different addresses to the same attacker.

We believe that AMMs are interested in leveraging the proposed metrics to build a detection system. Indeed, some have already put effort into this direction. For instance, PancakeSwap recently included in its interface a service called HashDit [33], which provides a risk level in investing in a liquidity pool. HashDit is a Token Contract Scanning service, that estimates the risk of a token by analyzing the code of its smart contract [8]. We believe the proposed metrics can enhance this and other existing services by adding insightful information.

10 Related Work

Tokens identification. In previous work, there are mainly two token identification approaches: behavior-based and interface-based. The behavior-based method assumes that a token contract maps addresses to the number of tokens owned and contains a function to transfer tokens. Chen et al. [13] follow this approach, analyzing the EVM execution path to find smart contracts data structures that indicate the bookkeeping of a token. The interface-based approach, the technique we take in this work, aims to find tokens that conform to specific interfaces (e.g., the ERC20 interface). This method involves discovering the implemented functions within the smart contract bytecode. Several works use this approach [14, 22, 63]. Frowis et al. [28] proved that the interface-based technique could detect 99% of the tokens in their ground truth dataset.

Liquidity pool scams. Xia et al. [66] characterize scam tokens on Ethereum. First, they leverage CoinMarketCap [18] to obtain a ground truth of official and scam tokens. They used The Graph [31] to obtain 21,778 tokens and 25,131 liquidity pools from May 2020 to December 2020. A guilt-by-association heuristic is adopted to enlarge the dataset, subsequently used to train a machine learning model. More than 11,182 fraudulent tokens were discovered after they ran their classifier on the expanded dataset. Mazorra et al. [43] extended Xia et al. dataset by including Uniswap data until 3 September 2021, discovering an additional 18 thousand scam tokens. They provide three categories for rug pulls: simple, sale, and trap-door. Then, they found that more than 97.7% of the tokens labeled as scams are involved in rug pulls.

Rug pull mitigation. Rug pulls are a very recent issue, and to the best of our knowledge there is no actual solution to prevent them. However, there is a new proposed standard and some protocols that can help to mitigate the problem. To counter the theft of tokens, Wang et al. [64] proposed a new token standard called ERC-20R. With this standard, a transaction is reversible for a short time (dispute period) after it has been performed. During this period, the sender can request to freeze the disputed asset to a set of decentralized judges. If judges agree to lock the disputed asset, it starts another period of time in which the sender can convince judges to revert the transaction. Instead, liquidity locker protocol (*e.g.,* Unicrypt [61]) allows locking LP-tokens inside smart contracts for a given amount of time. This solution assures that the liquidity cannot be removed from the pool until the timer expires, making rug pull impossible. Of course, this solution does not prevent rug pulls after the time expires or dumping one of the tokens in the liquidity pool.

11 Discussion

What is the impact of not collecting all the internal transactions? Unlike other works [14, 63], we do not collect all smart contracts generated by internal transactions. We collect

smart contracts created directly by EOAs, and expand our dataset by adding contracts that emitted at least one Transfer Event. This approach could lead to the loss of a small percentage of tokens. We can perform a rough estimation of the ERC-20 token we miss by comparing the number of tokens we retrieved with the number of tokens retrieved by Chen et al. [14] at the same block height. Our approach retrieves 146,928 tokens instead of 165,955, approximately 12% less. However, it is important to note that, by design, our approach misses only tokens that are never used, traded, or transferred. So, the missing tokens do not represent interesting cases for our study.

Why does it appear that rug pulls and token spammers are more frequent in BSC than in Ethereum? From a technical point of view, rug pulls work the same way in the two blockchains. Indeed, since BSC is EVM compliant and PancakeSwap is a fork of Uniswap, the same smart contract can be used on both blockchains. However, the cost of the operation is significantly different. As we saw in Sec. 7.1.1, performing a rug pull in BSC is cheaper (on average $10.5 with peaks of $600) than in Ethereum (on average $400 with peaks of over $2,000). These costs represent a fixed cost for the attacker, and going even or gaining money may be more difficult in Ethereum versus BSC.

Are cost-efficient blockchains vulnerable to 1-day rug pulls? As discussed, one of the possible reasons for the prevalence of rug pulls on BSC is the low transaction cost. This could suggest that cost-efficient blockchains are more vulnerable to 1-day rug pulls. However, to confirm this hypothesis, it is necessary to examine whether the phenomenon is common in blockchains with costs similar to the BSC.

Considering our case study of BSC, we believe the low cost of transactions is not the only reason for the high number of rug pulls. In particular, BSC provides one of the first DeFi ecosystems that is cheaper and faster than Ethereum. It quickly became very popular. Moreover, thanks to EVM compatibility, many no-code tools, libraries, and smart contracts already developed for Ethereum can also be used on BSC. This allows the deployment of smart contracts and the creation of new tokens with limited technical capabilities. Thus, the high number of potential victims, the little technical challenge, and the cost-efficiency made the BSC fertile ground for malicious actors to carry out 1-day rug pulls. Even though the low cost can facilitate rug pulls, increasing the costs of blockchains is not a real solution. Instead, a possibility is to shift the focus to DEXes's protocol and smart contracts for token creation. In particular, it could be possible to design more secure smart contracts to handle tokens (*e.g.,* ERC-20R) or AMM protocols with policies that disincentive rug pull operations.

Can different users coordinate to carry out the same operation, or can a user use multiple addresses? In this work, we considered each address belonging to a single different user, and we assumed there is no coordination among addresses. Nonetheless, a user may change the address he uses to perform each rug pull. It is also possible that a group of users coordinate to carry out the operation. For example, a user can create a liquidity pool while others perform wash trading. A possible approach to detect this malicious behavior is to gather all the transactions among the allegedly involved addresses and look for malicious patterns or communities (*e.g.,* using graph analysis). In this work, we do not perform this analysis, but we plan to explore more sophisticated rug pulls as an extension of this work.

12 Ethical considerations

In this paper, we examined 3 billion transactions from Ethereum and the BSC. We focused our research on the addresses that create tokens and how they use them. All data we retrieved is publicly available, and EOA addresses are pseudo-anonymous. We never attempted to deanonymize the addresses or violate their privacy during this work. Consequently, and in accordance with our IRB's policies, we did not require express approval to conduct our analysis.

13 Conclusion and Future Work

In this work, we conduct a thorough investigation of the tokens and the liquidity pools of the BNB Smart Chain and Ethereum. We studied the lifetime of the tokens and their creators. We discovered two very interesting metrics: 60% of the total tokens of both blockchains do not survive their first day (1-day token), and a tiny fraction of addresses (1% of addresses), which we called token spammers, created more than 20% of the tokens. We explore the correlation between token spammers and 1-day tokens, and we found that token spammers strongly impact the existence of 1-day tokens.

More interestingly, we find that token spammers use 1-day tokens as disposable tokens to arrange rug pulls, exploiting the mechanism of liquidity pools. We selected from our dataset all the liquidity pools that show evidence of a rug pull and dissect the operations, analyzing them from several perspectives. Finally, we introduce the sniper bot, trading bot that aims to buy tokens at their listing price. However, they unwillingly became victims of the rug pulls because of their mechanism.

As future work, we believe it is interesting to further refine our results by including addresses that cooperate to perpetrate rug pulls in the analysis. It could be possible to uncover other malicious and more sophisticated patterns. As discussed in Sec. 11, cost-efficient blockchains could be more exposed to the 1-day rug pulls. Thus, it is interesting to extend our analysis to blockchains with transaction costs comparable to BSC (*e.g.,* Algorand [30]). Finally, another promising direction is further exploring sniper bots to provide a more detailed analysis of their typologies and operations.

References

[1] Hayden Adams, Noah Zinsmeister, and Dan Robinson. Uniswap v2 core. 2020.

[2] adamsnipes. Pancakeswap bot & uniswap bot. https://adamsnipes.io/home.html, 2022.

[3] Osato Avan-Nomayo. Pancakeswap dex reportedly set to block users from iran. https://www.theblockcrypto.com/linked/133904/pancakeswap-dex-reportedly-set-to-block-users-from-iran, 2022.

[4] Massimo Bartoletti, Salvatore Carta, Tiziana Cimoli, and Roberto Saia. Dissecting ponzi schemes on ethereum: identification, analysis, and impact. *Future Generation Computer Systems*, 102:259–277, 2020.

[5] Dirk G Baur and Thomas Dimpfl. Asymmetric volatility in cryptocurrencies. *Economics Letters*, 173:148–151, 2018.

[6] Binance. Binance chain docs - json-rpc endpoint. https://docs.binance.org/smart-chain/developer/rpc.html, 2022.

[7] Binance. Bnb chain documentation. https://docs.bnbchain.world/docs/learn/intro, 2022.

[8] Binance. Pancakeswap integrates token contract scanning directly on its swap page. https://www.binance.com/en/news/flash/7193825, 2022.

[9] Binance. Proof of authority explained. https://academy.binance.com/en/articles/proof-of-authority-explained, 2022.

[10] Steven Brock. Scalping in ecommerce: Ethics and impacts. *Available at SSRN 3793357*, 2021.

[11] Vitalik Buterin et al. A next-generation smart contract and decentralized application platform. *white paper*, 3(37), 2014.

[12] Yi Cao, Yuhua Li, Sonya Coleman, Ammar Belatreche, and Thomas Martin McGinnity. Detecting wash trade in financial market using digraphs and dynamic programming. *IEEE transactions on neural networks and learning systems*, 27(11):2351–2363, 2015.

[13] Ting Chen, Yufei Zhang, Zihao Li, Xiapu Luo, Ting Wang, Rong Cao, Xiuzhuo Xiao, and Xiaosong Zhang. Tokenscope: Automatically detecting inconsistent behaviors of cryptocurrency tokens in ethereum. In *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*, pages 1503–1520, 2019.

[14] Weili Chen, Tuo Zhang, Zhiguang Chen, Zibin Zheng, and Yutong Lu. Traveling the token world: A graph analysis of ethereum erc20 token ecosystem. In *Proceedings of The Web Conference 2020*, pages 1411–1421, 2020.

[15] Weimin Chen, Xinran Li, Yuting Sui, Ningyu He, Haoyu Wang, Lei Wu, and Xiapu Luo. Sadponzi: Detecting and characterizing ponzi schemes in ethereum smart contracts. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 5(2):1–30, 2021.

[16] CoinGecko. Coingecko api. https://www.coingecko.com/en/api, 2022.

[17] Patrick Thompson CoinGeek. Solana sees first rug pull: Luna yield disappears with $6.7m in digital currency. https://coingeek.com/solana-sees-first-rug-pull-luna-yield-disappears-with-6-7m-in-digital-currency/, 2021.

[18] CoinMarketCap. Coinmarketcap. https://coinmarketcap.com/, 2022.

[19] Yashu Gola Cointelegraph. Game over! squid game-inspired crypto scam collapses as price crashes from $2.8k to zero. https://cointelegraph.com/news/game-over-squid-game-inspired-crypto-scam-collapses-as-price-crashes-from-2-8k-to-zero, 2021.

[20] damartripamungkas. Botdexdamar. https://github.com/damartripamungkas/botdexdamar, 2022.

[21] Chris Dannen. *Introducing Ethereum and solidity*, volume 1. Springer, 2017.

[22] Monika Di Angelo and Gernot Salzer. Identification of token contracts on ethereum: standard compliance and beyond. *International Journal of Data Science and Analytics*, pages 1–20, 2021.

[23] Morris J Dworkin et al. Sha-3 standard: Permutation-based hash and extendable-output functions. 2015.

[24] Ethereum. Contract abi specification. https://docs.soliditylang.org/en/v0.8.13/abi-spec.html, 2022.

[25] Ethereum. Ethereum virtual machine (evm). https://ethereum.org/it/developers/docs/evm/, 2022.

[26] Vitalik Buterin Fabian Vogelsteller. Eip-20: Token standard. https://eips.ethereum.org/EIPS/eip-20, 2015.

[27] Fantom Foundation. Fantom whitepaper. https://fantom.foundation/research/wp_fantom_v1.6.pdf, 2022.

[28] Michael Fröwis, Andreas Fuchs, and Rainer Böhme. Detecting token systems on ethereum. In *International conference on financial cryptography and data security*, pages 93–112. Springer, 2019.

[29] Bingyu Gao, Haoyu Wang, Pengcheng Xia, Siwei Wu, Yajin Zhou, Xiapu Luo, and Gareth Tyson. Tracking counterfeit cryptocurrency end-to-end. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 4(3):1–28, 2020.

[30] Yossi Gilad, Rotem Hemo, Silvio Micali, Georgios Vlachos, and Nickolai Zeldovich. Algorand: Scaling byzantine agreements for cryptocurrencies. In *Proceedings of the 26th symposium on operating systems principles*, pages 51–68, 2017.

[31] The Graph. The graph: Apis for a vibrant decentralized future. https://thegraph.com/en/, 2022.

[32] Martin Haferkorn and Josué Manuel Quintana Diaz. Seasonality and interconnectivity within cryptocurrencies-an analysis on the basis of bitcoin, litecoin and namecoin. In *International Workshop on Enterprise Applications and Services in the Finance Industry*, pages 106–120. Springer, 2014.

[33] HashDit. Risk level description. https://hashdit.github.io/hashdit/docs/risk-level-description, 2023.

[34] Infura. Infura. https://infura.io/, 2022.

[35] Don Johnson, Alfred Menezes, and Scott Vanstone. The elliptic curve digital signature algorithm (ecdsa). *International journal of information security*, 1(1):36–63, 2001.

[36] P.C. Kotsias. pcko1/etherscan-python. https://github.com/pcko1/etherscan-python, 2020.

[37] P.C. Kotsias. pcko1/bscscan-python. https://github.com/pcko1/bscscan-python, 2021.

[38] Abhinandan Kulal. Followness of altcoins in the dominance of bitcoin: A phase analysis. *Macro Management & Public Policies*, 3(3), 2021.

[39] Massimo La Morgia, Alessandro Mei, Francesco Sassi, and Julinda Stefa. Pump and dumps in the bitcoin era: Real time detection of cryptocurrency market manipulations. In *2020 29th International Conference on Computer Communications and Networks (ICCCN)*, pages 1–9. IEEE, 2020.

[40] Solidity Lang. Contract abi specification. https://docs.soliditylang.org/en/v0.5.3/abi-spec.html, 2022.

[41] Defi Llama. Defi llama. https://defillama.com/, 2022.

[42] Youcef Maouchi, Lanouar Charfeddine, and Ghassen El Montasser. Understanding digital bubbles amidst the covid-19 pandemic: Evidence from defi and nfts. *Finance Research Letters*, 47:102584, 2022.

[43] Bruno Mazorra, Victor Adan, and Vanesa Daza. Do not rug on me: Leveraging machine learning techniques for automated scam detection. *Mathematics*, 10(6):949, 2022.

[44] Evgeny Medvedev and the D5 team. Ethereum etl. https://github.com/blockchain-etl/ethereum-etl, 2018.

[45] Massimo La Morgia, Alessandro Mei, Francesco Sassi, and Julinda Stefa. The doge of wall street: Analysis and detection of pump and dump cryptocurrency manipulations. *ACM Transactions on Internet Technology (TOIT)*, 2021.

[46] Jason Carve Piper Merriam. Web3.py. https://web3py.readthedocs.io/en/stable/, 2022.

[47] Amy Cheng The Washington Post. 'squid game'-inspired cryptocurrency that soared by 23 million percent now worthless after apparent scam. https://www.washingtonpost.com/world/2021/11/02/squid-game-crypto-rug-pull/, 2021.

[48] Valerio Puggioni. Crypto rug pulls: What is a rug pull in crypto and 6 ways to spot it. https://cointelegraph.com/explained/crypto-rug-pulls-what-is-a-rug-pull-in-crypto-and-6-ways-to-spot-it, 2022.

[49] PumpBot. Sniper bot crypto: Chain sniper- the all in one sniper bot, dex bot, pinksale bot. https://pump-bot.com/crypto-bots/sniper-bot-frontrunner-chainsniper-dexbot, 2022.

[50] Kaihua Qin, Liyi Zhou, and Arthur Gervais. Quantifying blockchain extractable value: How dark is the forest? In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 198–214. IEEE, 2022.

[51] Brian Rudick. Defi summer 2.0: Don't call it a comeback. https://www.gsr.io/insights/chart-of-the-week-defi-summer-2-0-dont-call-it-a-comeback/, 2021.

[52] saantiaguilera. Ax-50 liquidity sniper. https://github.com/saantiaguilera/liquidity-sniper, 2022.

[53] Kevin Sekniqi, Daniel Laine, Stephen Buttolph, and Emin G¨un Sirer. *Avalanche Platform*, volume 1. online, 2020.

[54] Arijit Sarkar The New York State Senate. The secret to the success of 'squid game,' explained. https://www.forbes.com/sites/danidiplacido/2021/10/06/the-secret-to-squid-games-success-explained/?sh=83a56ea224cf, 2021.

[55] Arijit Sarkar The New York State Senate. Ny sen. thomas proposes to criminalize rug pulls and other crypto frauds. https://www.nysenate.gov/newsroom/in-the-news/kevin-thomas/ny-sen-thomas-proposes-criminalize-rug-pulls-and-other-crypto, 2022.

[56] Eva Su. *Digital Assets and SEC Regulation*. Congressional Research Service, 2020.

[57] Martin Holst Swende and Marius van der Wijden. Eip-3155: Evm trace specification. https://eips.ethereum.org/EIPS/eip-3155, 2022.

[58] Christof Ferreira Torres, Ramiro Camino, et al. Frontrunner jones and the raiders of the dark forest: An empirical study of frontrunning on the ethereum blockchain. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 1343–1359, 2021.

[59] Christof Ferreira Torres, Mathis Steichen, et al. The art of the scam: Demystifying honeypots in ethereum smart contracts. In *28th USENIX Security Symposium (USENIX Security 19)*, pages 1591–1607, 2019.

[60] Trading-Tiger. Pancakeswap bsc sniper bot. https://github.com/Trading-Tiger/Pancakeswap_BSC_Sniper_Bot, 2022.

[61] UniCrypt. Liquidity lockers - unicrypt. https://docs.unicrypt.network/liquidity-lockers/general-concept, 2023.

[62] Uniswap. Uniswap v2 license. https://github.com/Uniswap/v2-core/blob/master/LICENSE, 2022.

[63] Friedhelm Victor and Bianca Katharina Lüders. Measuring ethereum-based erc20 token networks. In *International Conference on Financial Cryptography and Data Security*, pages 113–129. Springer, 2019.

[64] Kaili Wang, Qinchen Wang, and Dan Boneh. Erc-20r and erc-721r: Reversible transactions on ethereum. *arXiv preprint arXiv:2208.00543*, 2022.

[65] Gavin Wood. Ethereum yellow paper: A formal specification of ethereum, a programmable blockchain. 2018. *URL https://github. com/ethereum/yellowpaper*, 2018.

[66] Pengcheng Xia, Haoyu Wang, Bingyu Gao, Weihang Su, Zhou Yu, Xiapu Luo, Chao Zhang, Xusheng Xiao, and Guoai Xu. Trade or trick? detecting and characterizing scam tokens on uniswap decentralized exchange. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 5(3):1–26, 2021.

[67] Dirk A Zetzsche, Douglas W Arner, and Ross P Buckley. Decentralized finance. *Journal of Financial Regulation*, 6(2):172–203, 2020.

Table 4: Comparison with other blockchain operations. We report only on frauds performed on Ethereum since our work is the first to analyze the Binance Smart Chain.

Operation	Tot Gain ($)	# Addresses	# Operations	Blockchain	From	To
[29] Counterfeit Tokens	17.35M	364	573	Ethereum	2015-07-30	2020-03-18
[58] Displacement	4.1M	74	2,983	Ethereum	2015-07-30	2020-11-21
[50] Fixed Spread Liquidations	89.18M	2,724	31,057	Ethereum	2018-12-01	2021-08-05
[59] Honeypots	90K	53	690	Ethereum	2015-08-07	2018-10-12
[58] Insertion	13.9M	1,975	196,691	Ethereum	2015-07-30	2020-11-21
[50] Sandwich Attacks	174.34M	3,488	750,529	Ethereum	2018-12-01	2021-08-05
[15] Smart Ponzi	17.70M	444	835	Ethereum	2015-08-01	2020-05-20
[58] Suppression	1.03M	128	50	Ethereum	2015-07-30	2020-11-21
1-day rug pulls	148.93M	16,439	21,594	Ethereum	2015-07-30	2022-03-07
1-day rug pulls	90.78M	117,110	266,340	BSC	2020-04-20	2022-03-07

A Functions and events of the ERC-20 and BEP-20 standards

Table 5: Functions and events of the ERC-20 (Ethereum) and BEP-20 (Binance Smart Chain) standard interface. We report in yellow the methods that are optional in the ERC-20 interface and in red the only method that is optional in both interfaces.

Function	Signature
name()	06fdde03
symbol()	95d89b41
decimals()	313ce567
totalSupply()	18160ddd
balanceOf(address)	70a08231
transfer(address,uint256)	a9059cbb
transferFrom(address,address,uint256)	23b872dd
approve(address,uint256)	095ea7b3
allowance(address,address)	dd62ed3e
Event	**Signature**
Transfer(address,address,uint256)	ddf252ad
Approval(address,address,uint256)	095ea7b3

B Are 1-day rug pulls frauds?

1-day rug pulls are very different from more notorious rug pulls like Squid Game [47] or Luna Yield [17]. Indeed, these operations lasted weeks or months, and their perpetrator exploited extensive marketing campaigns and misleading advertising to deceive users into investing in their tokens. In the case of Squid Game, the scammer created a token in the BSC following the success of the homonym Netflix television series [54]. Due to the extensive marketing campaign promoting the token as official on social media platforms such as Twitter and Telegram, its value skyrocketed from a few cents to over $2,856 in less than a week [19]. Then, the scammer removed nearly all of the liquidity from the pool ($3.3 million), causing the token's value to plummet to near zero [47]. In our paper, we study 1-day operations that aim to make a profit with the least possible effort in a short time frame. For this reason, it is unlikely that they leverage sophisticated marketing campaigns to lure investors, like in the case of Squid Game. However, some 1-day rug pull operations use other kinds of deceptive tactics. The first uses token names identical or slightly different from well-known companies or popular tokens. As we saw in Sec. 7.2.3, this case involves 8.7% of Ethereum rug pulls, and 10% of BSC rug pulls. Another deceptive technique consists in attempting to legitimate the project by verifying the smart contract code on BSCscan and Etherscan. The verification consists in uploading the source code so that the platform can compile it and verify that it matches the bytecode of the token stored in the blockchain. The verification provides users transparency and gives more guarantee that the token is not fraudulent. We find the smart contract is available and verified for 55% (147,069) of BSC and 67% (14,722) of Ethereum tokens involved in the 1-day rug pull operations. Finally, another technique to legitimate the token consists in creating the "official" Telegram group of the token. We find evidence of this technique in the smart contract's code and then inspect the groups on Telegram. Indeed, analyzing the source codes, we notice that 19,096 token smart contracts in BSC and 1,334 in Ethereum report a link to the Telegram group of the token. Although the organizers of 1-day rug pulls use deceptive techniques to dupe investors, we cannot consider these operations frauds because the phenomenon is still not regulated. In any case, people lose money: Investors bought the token in 92.7% of the Ethereum rug pulls and in 91.2% of the BSC ones, and in all these cases the investment is lost. For this reason, we believe that these operations, even if not illegal, are exploitative of the DeFi ecosystem and should be contrasted to safeguard investors. Indeed, regulators are starting to take action to contrast them.

For example, New York State Senator Kevin Thomas proposes criminalizing rug pulls and other crypto frauds by introducing a new bill amendment request (Senate Bill S8839) [55]. The idea of the bill is to introduce the crime of *illegal rug pull* that occurs if the creator of the token sells more than 10% of his tokens within five years of their last sale.

C Token names

Table 6: Token names most frequently used in 1-day rug pull operations.

BNB Smart Chain		Ethereum	
Name	# of tokens	Name	# of tokens
Pornhub	1,023	Hyve.works	50
Galaxy	588	Deriswap	32
Seedswap	502	Shibaswap	28
Lionswap	429	Apple core finance	17
Eco.finance	421	X20.finance	16
Spacex	419	Yield farm rice	15
Onlyfans	419	The sandbox	14