



SAPIENZA  
UNIVERSITÀ DI ROMA

# Airdrop-Based Threats on Ethereum: Analysis and Detection

Faculty of Engineering, Computer Science and Statistics  
Master Degree in Computer Science

Candidate

Giovanni Montobbio  
ID number 1845035

Thesis Advisor

Prof. Alessandro Mei

Co-Advisor

Prof. Eugenio Nemmi

Academic Year 2023/2024

Thesis not yet defended

---

**Airdrop-Based Threats on Ethereum: Analysis and Detection**

Master thesis. Sapienza – University of Rome

© 2024 Giovanni Montobbio. All rights reserved

This thesis has been typeset by  $\text{\LaTeX}$  and the Sapthesis class.

Version: October 25, 2024

Author's email: [montobbio.1845035@studenti.uniroma1.it](mailto:montobbio.1845035@studenti.uniroma1.it)

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Background</b>	<b>3</b>
2.1	Blockchain . . . . .	3
2.1.1	Consensus algorithms . . . . .	3
2.1.2	Types of Blockchains . . . . .	4
2.1.3	Application domain . . . . .	6
2.2	Ethereum . . . . .	6
2.2.1	Smart contracts . . . . .	7
2.2.2	Ethereum Virtual Machine . . . . .	7
2.2.3	Tokens . . . . .	8
2.2.4	Transactions . . . . .	8
2.2.5	Airdrop . . . . .	10
2.3	Scams . . . . .	11
2.3.1	Phishing . . . . .	11
2.3.2	Rug Pulls . . . . .	11
2.3.3	Airdrop Scams . . . . .	12
<b>3</b>	<b>State of the art</b>	<b>13</b>
3.1	Machine Learning . . . . .	13
3.2	Related works . . . . .	13
<b>4</b>	<b>Datasets</b>	<b>16</b>
4.1	Tokens dataset . . . . .	16
4.2	Etherscan . . . . .	16
4.2.1	Etherscan APIs . . . . .	16
4.3	Transactions dataset . . . . .	17
4.4	Transfers and holders dataset . . . . .	21
4.5	Differences between the creation and the first event and between the first and last event . . . . .	22
4.6	Senders and receivers . . . . .	24
4.7	Ground truths . . . . .	25
4.7.1	Metamask eth-phishing-detect . . . . .	26
4.7.2	Phishfort . . . . .	26
4.7.3	Forta network . . . . .	27
4.7.4	One day tokens . . . . .	28

---

4.7.5	Webacy . . . . .	28
4.7.6	Non malicious airdrops . . . . .	29
4.7.7	Manual labeling . . . . .	29
<b>5</b>	<b>Methodology</b>	<b>34</b>
5.1	Features . . . . .	34
5.1.1	Chargrams . . . . .	35
5.1.2	Feature scaling . . . . .	36
5.2	Models . . . . .	38
5.2.1	Random Forest . . . . .	38
5.2.2	SVM . . . . .	39
5.2.3	Gradient boosting . . . . .	39
<b>6</b>	<b>Test &amp; Results</b>	<b>41</b>
6.1	Random Forest . . . . .	41
6.2	SVM . . . . .	42
6.3	Gradient boosting . . . . .	43
<b>7</b>	<b>Conclusions</b>	<b>45</b>

# Chapter 1

## Introduction

Blockchain technology represents an innovative approach in the technological landscape. Among the most important blockchains, Ethereum has emerged as one of the most prominent platforms, enabling the creation of decentralized applications (DApps) and smart contracts. Ethereum has a rapidly evolving ecosystem where users process millions of transactions and create numerous applications every day. However, the rapid growth of blockchain and cryptocurrency has also attracted malicious actors, leading to the rise of new forms of cyber threats and scams. This increase makes developing methods to detect and counter those threats essential.

This phenomenon is so widespread that numerous studies have focused on the various threats present on the blockchain. Among the new tools introduced by blockchain technology, there is a new distribution method called "Airdrop", a mass distribution of currencies or assets to a wide number of addresses. Indeed, malicious actors find a way to exploit airdrops to create new threats that we call "Airdrop scams". There is no specific work on this type of threat, so the goal we set ourselves is intriguing but uncharted.

In this thesis we explore the landscape of airdrop-based threats on the Ethereum network, focusing on how these schemes work and try to find common patterns in those activities and propose detection strategies.

We start by collecting information about ERC-20 Ethereum tokens and then create a dataset containing different information about the tokens and their transactions.

Some of the information we are interesting in are: the number of transactions, transfers, and holders; the lifespan of the token; the name of the token; the number of addresses that sent the token and the number that received the token; the total supply of the token; the quantity, average and standard deviation of tokens sent in a single transaction.

Finally, we construct a dataset of Ethereum tokens that were distributed using the airdrop method and apply a machine learning algorithm to identify common patterns in airdrop-based scams for detection purposes.

Since it doesn't exist a dataset from which is possible to retrieve labels for the malicious airdrops, as part of the thesis work, we gathered information using different sources and created a new labeled dataset. This is one of the different challenges we face in our task, the finding of the ground truths, the labels for the tokens. Since

there is no single source for this information, we rely on various tools to collect the labels, among which we perform manual searches.

The final dataset contains about 1000 labelled samples, and 16 features. Finally, we use this dataset to train three different Machine Learning models: Random Forest, SVM and Gradient Boosting. Our tests show that all these models can achieve good performances. Overall we were able to achieve the best result with the Gradient Boosting, with an accuracy of 98% and a score of 99% for precision, recall and f1-score.

Although much more work can be done in this area, this research serves as a valuable first step in analyzing the phenomenon of airdrop-based scams. We believe it can raise awareness within the community and lay the groundwork for future studies in this field.

## Chapter 2

# Background

### 2.1 Blockchain

Blockchain [20] is a technology that creates a distributed, immutable, and secure ledger of transactions, maintaining a shared list of records. These records, called blocks, contain encrypted data on the history of every previous block, with precise timestamped transaction information. The blocks are linked together, forming the blockchain.

A blockchain consists of two main components: a decentralized network that facilitates and verifies transactions, and the immutable ledger maintained by that network. Every participant in the network can see this shared transaction ledger, but there is no single point of failure that could be hacked or corrupted.

Thanks to the integrated ledger, each transaction is recorded only once, eliminating the risk of double spending. Additionally, once a transaction is recorded, it cannot be altered, making all transactions immutable.

When a transaction is made, it is signed with a public key to ensure the security of the blockchain. Validation is achieved through a decentralized consensus protocol [5], which adds new blocks to the chain. The most common protocols are Proof of Stake (PoS) [6] and Proof of Work (PoW) [7]. While all transactions are visible to network participants, cryptography is used to ensure privacy by maintaining user anonymity.

#### 2.1.1 Consensus algorithms

Consensus mechanisms [4] [16] are incorporated into blockchains to provide a fault-tolerant method for transaction verification. Consensus ensures that nodes in the network reach an agreement. As the network grows, the number of nodes increases, making it more challenging to achieve this agreement. Since blockchain is a dynamic and self-regulating system, a secure mechanism is necessary to guarantee transaction authenticity. Various consensus mechanisms have been proposed, each with different principles and applications.

- **Proof of work** [7]: Proof-of-work (PoW) is a consensus algorithm used to validate blocks. The purpose of PoW is to prevent malicious actors from altering the blockchain's historical data and to add new blocks to the blockchain

in a secure and distributed manner. The miners, participants of the blockchain who use computational power to validate and add new transactions to the blockchain, compete to solve a cryptographic puzzle, which is a mathematical problem that is difficult to solve but easy to verify. The first node to solve the puzzle wins and shares the new block with the rest of the network and is rewarded with a certain amount of the network's cryptocurrency. Solving the puzzle creates the cryptographic link between the current block and the previous one. The canonical chain is determined by a fork-choice rule that selects the blocks with the most work invested in mining them. The system remains secure because defrauding the chain would require 51% of the network's computing power, an investment so large in equipment and energy that it outweighs potential gains.

- **Proof of stake** [6]: In PoS blockchains blocks are created by validators. Validators are nodes that are selected to propose a new block. They are responsible for checking that new blocks propagated over the network are valid and occasionally creating and propagating new blocks themselves. In each slot, a validator is randomly selected to propose a block. Their consensus client requests a set of transactions, called an "execution payload", from the execution client. The validator wraps these transactions with consensus data to form a block, which is then broadcasted to other nodes on the Ethereum network. Validators earn rewards in cryptocurrency for creating blocks. In rare cases where multiple blocks are proposed for the same slot, or when nodes receive blocks at different times, the fork-choice algorithm selects the block that forms the chain with the highest attestation weight. Attestation weight refers to the number of validators supporting a block, scaled by their stake (balance). PoS is secure because an attacker would need to destroy a large amount of ETH to compromise the chain. Rewards encourage honest behavior, while penalties discourage malicious actions.

Both algorithms offer strong security. However, PoW consumes significantly more computational power since it uses a hardware-based algorithm. Conversely, PoS requires a substantial initial investment, which allows key stakeholders in the system to influence the random selection of validators.

### 2.1.2 Types of Blockchains

A primary distinction in blockchain technology is between permissionless and permissioned blockchains. In a permissionless blockchain, anyone can join the network and participate in activities without needing additional permissions, and users are pseudo-anonymous. In a permissioned blockchain, access is restricted, and users must be accepted before joining. Within the network, the identity of all participants is known. Permissioned blockchains are primarily related to the company world. There are four main types of blockchain networks: public blockchains, private blockchains, consortium blockchains, and hybrid blockchains. Each type has its advantages, disadvantages, and ideal use cases.

- **Public:** This is where cryptocurrencies like Bitcoin [19] originated, popularizing distributed ledger technology (DLT). It eliminates the issues of



centralization, such as reduced security and transparency. Public blockchains are non-restrictive and permissionless, allowing anyone with internet access to become a node. Users can access records, conduct mining activities and verify transactions. Public blockchains are independent of any organization, so they can continue to operate even if the founding organization ceases to exist, as long as computers remain connected to the network. However, public blockchains can be slow and do not scale well as more nodes join the network. Common use cases include digital currencies like Bitcoin and Ethereum; Decentralized finance (DeFi) platforms which offer services like trading, lending and borrowing without the need for traditional banks or financial institutions; supply chain tracking, tracking and verifying the provenance of goods in a supply chain, ensuring transparency and authenticity.

- **Private:** A private blockchain operates in a closed network controlled by a single entity. While it uses peer-to-peer connections and decentralization, it functions on a smaller scale. Unlike public blockchains, where anyone can join, private blockchains restrict participation and are often referred to as permissioned or enterprise blockchains. The controlling organization sets permission levels, security, and access controls.

Private blockchains are faster and process transactions more efficiently than public ones, but they are criticized for lacking true decentralization, a core principle of blockchain. Trust in the information may be lower because centralized nodes control the validity of data. The small number of nodes can also pose security risks, and rogue nodes could compromise the consensus. Additionally, there is no anonymity in private blockchains. They are often used in internal voting systems or supply chain management.

- **Hybrid:** Hybrid blockchains combine elements of both public and private blockchains. They allow organizations to create a private, permissioned system alongside a public, permissionless one. This enables control over who can access specific data while making other data publicly accessible. Typically, transactions in a hybrid blockchain are private but can be verified when necessary, for example, through smart contracts, immutable pieces of code that run on a blockchain when invoked by some transaction.

A hybrid blockchain protects privacy while allowing communication with third parties. It offers better scalability and faster transactions than a public blockchain and is less vulnerable to a 51% attack. However, it lacks full transparency, and upgrading can be challenging. Since there is no incentive for participants, user contribution may be lower.

- **Consortium** [26]: Similar to hybrid blockchains, consortium blockchains mix private and public elements. However, they are controlled by a group of organizations rather than a single entity. In a consortium blockchain, multiple members collaborate on a decentralized network. Consensus is managed by preselected nodes. Validator nodes initiate, receive, and validate transactions, while member nodes can initiate or receive them.

Consortium blockchains offer more security, scalability, and efficiency than

public blockchains. They also provide access control similar to private and hybrid blockchains. However, they are less transparent than public blockchains and can be compromised if a member node is breached.

### 2.1.3 Application domain

The blockchain has a wide range of applications [12], from cryptocurrencies to supply chain management, from electronic voting to decentralized finance (DeFi) applications.

- **Financial Sector:** Blockchain enables faster, more secure transactions by removing intermediaries and reducing costs. It supports cross-border payments, smart contracts, decentralized lending, promoting financial inclusion and innovation.
- **Supply Chain Management:** Blockchain provides visibility and traceability across the entire supply chain. It allows tracking and verification of products at every stage, ensuring authenticity, preventing counterfeiting, and building trust. Blockchain can also optimize logistics and improve inventory management.
- **Healthcare:** Blockchain addresses challenges like data interoperability and patient privacy. It securely stores and shares medical records, ensuring data integrity and enabling seamless information exchange. Blockchain can also verify the authenticity of pharmaceutical products, enhancing patient safety.
- **Identity Management:** Traditional systems are vulnerable to data breaches and identity theft. Blockchain offers a decentralized, tamper-proof platform for identity verification. It gives individuals control over their personal data, enhancing privacy and reducing the risk of identity fraud.
- **Voting Systems:** Blockchain improves electoral processes by ensuring transparent, tamper-proof voting. It prevents duplicate votes, securely verifies voter identities, and provides an auditable trail, increasing trust in the democratic process.
- **Intellectual Property:** Blockchain helps protect intellectual property rights by providing a secure, transparent platform for registering and managing assets like patents and trademarks. It offers proof of ownership, prevents unauthorized use, and facilitates fair royalty distribution.
- **Energy Sector:** Blockchain supports peer-to-peer energy trading and renewable energy tracking. It allows direct transactions between producers and consumers, reducing costs by eliminating intermediaries. It also promotes transparency in energy sourcing and incentivizes sustainable practices.

## 2.2 Ethereum

Ethereum [2] is a blockchain technology that powers the cryptocurrency Ether (ETH). It operates on a Proof-of-Stake consensus algorithm and allows the deployment

of smart contracts. These are decentralized pieces of code executed directly on the blockchain, contributing to Ethereum's success. Smart contracts enable the development of decentralized applications (DApps) and are fundamental to the evolution of Web 3.0.

### 2.2.1 Smart contracts

Smart contracts [13] are self-executing computer programs stored on a blockchain. When they are invoked by a transaction that satisfies some predetermined conditions. Once deployed on the blockchain, a smart contract operates independently, and the blockchain is updated once the transaction is completed. This process is irreversible, and only authorized parties can view the results.

Smart contracts are written in programming languages like Solidity and run on the Ethereum Virtual Machine (EVM).

The advantages of smart contracts include:

- Speed, efficiency and accuracy: Once it is called by a transaction that satisfies a condition, the contract executes immediately. Being digital and automated, smart contracts eliminate paperwork and reduce errors that come with manual processing.
- Trust and transparency: With no intermediaries involved and shared encrypted records, participants can trust that the information has not been tampered with.
- Savings: Smart contracts eliminate intermediaries, reducing time delays and associated fees.

### 2.2.2 Ethereum Virtual Machine

The Ethereum Virtual Machine (EVM) [8] is a key component of the Ethereum blockchain. It is a decentralized virtual environment that executes code consistently and securely across all Ethereum nodes. Nodes run the EVM to execute smart contracts, using "gas" to measure the computational effort required for operations, ensuring efficient resource allocation and network security. Instead of a distributed ledger, Ethereum is a distributed state machine. Ethereum's state is a large data structure which holds not only all accounts and balances, but a machine state, which can change from block to block according to a pre-defined set of rules, and which can execute arbitrary machine code. The specific rules of changing state from block to block are defined by the EVM. The EVM behaves as a mathematical function would: Given an input, it produces a deterministic output. It therefore is quite helpful to more formally describe Ethereum as having a state transition function:

$$Y(S, T) = S'$$

Given an old valid state (S) and a new set of valid transactions (T), the Ethereum state transition function  $Y(S, T)$  produces a new valid output state  $S'$

The EVM is instrumental in maintaining consensus across the Ethereum blockchain. Every node in the Ethereum network runs the EVM, ensuring that all nodes agree on the state of the blockchain. This consensus is vital for the security and integrity of the Ethereum network.

### 2.2.3 Tokens

Tokens are cryptoassets that run on top of another cryptocurrency's blockchain. Generally, crypto tokens serve a specific purpose defined by the organization or individual that created them. This purpose can range from raising funds to providing access to services. Like other crypto assets, tokens are decentralized and use cryptographic signatures for security and record-keeping. However, cryptocurrencies are integrated into the blockchain itself, tokens are governed by smart contracts.

Tokens serve a multitude of purposes:

- **Economic Value Representation:** Tokens represent economic value and act as a currency within the system. Users can exchange tokens for goods, services, or other assets within the decentralized network.
- **Rights and Rules Definition:** Tokens can define rights and establish rules within the decentralized system. They regulate access, permissions, and privileges for various users or entities.
- **Representation of Digital and Non-Functional Assets:** Tokens can signify digital or non-fungible assets. A notable example is Non-Fungible Tokens (NFTs), which represent unique works of art or items. NFTs demonstrate digital ownership and provenance of singular, indivisible items, distinguishing them from fungible tokens.

#### 2.2.3.1 Token ERC Standard

Ethereum smart contracts can be used to create different types of tokens, each of which has specific functions and uses.

- **ERC-20 tokens:** These are the most popular Ethereum tokens and the standard for token creation on the platform. ERC-20 tokens are interchangeable and adhere to an interoperability standard, making them easy to implement in various DApps and trade on decentralized marketplaces. They often represent units of resources, such as loyalty points or gaming tokens.
- **ERC-721 tokens:** These are tokens that represent unique, non-fungible assets, such as works of art or real estate. They signify goods with intrinsic value or those that are rare or unique. We will analyze them in greater detail later.
- **ERC-1155 tokens:** These tokens combine the features of ERC-20 and ERC-721, allowing management of both fungible and non-fungible assets within a single contract. ERC-1155 tokens are especially suited for decentralized games or collections of assets. They are often used as tickets and adapt well to the needs of both ERC-20 and ERC-721 tokens.

### 2.2.4 Transactions

Ethereum transactions are the fundamental operations that allow users to interact with the Ethereum network. Differently from traditional transactions, which typically

involve only the transfer of value, Ethereum transactions can also trigger the execution of smart contracts, making them more versatile and complex.

An Ethereum transaction is a signed data package that contains the following key components:

- **From:** denotes the address of the sender
- **Recipient:** This field can represent another user's address for a value transfer or a smart contract's address for contract execution.
- **Nonce:** A counter that represents the number of transactions sent from a particular address. It ensures that each transaction is unique and prevents replay attacks.
- **Value:** The amount of Ether (ETH) to be transferred in the transaction.
- **Gas Price:** The amount of Ether the sender is willing to pay per unit of gas. Gas is the computational resource required to execute the transaction.
- **Gas Limit:** The maximum amount of gas that the sender is willing to consume for the transaction. It serves as a cap to ensure that transactions do not run indefinitely.
- **Data:** A field that contains the input data or instructions for executing a smart contract. For simple ETH transfers, this field is often empty.
- **V, R, S:** Cryptographic components derived from the sender's private key, used to create a digital signature. This signature ensures the transaction's authenticity and integrity.

#### 2.2.4.1 Types of transactions

Ethereum transactions can be categorized into several types based on their purpose and the data they carry.

- **Standard ETH Transfer:** The most basic type of transaction, a certain amount of Ether is sent from one address to another.
- **Contract Deployment:** A special type of transaction where the data field contains the bytecode of a smart contract. This transaction deploys a new smart contract on the Ethereum network.
- **Smart Contract Execution:** Transactions that interact with an existing smart contract by calling one of its functions. The data field includes the encoded function signature and any input parameters required for execution.
- **Token Transfer:** A type of smart contract execution transaction that involves transferring tokens from one address to another. These transactions invoke the transfer function of the token's smart contract.

### 2.2.5 Airdrop

The blockchain revolution has modified the landscape of finance and technology and has also introduced new ways of distributing digital assets. One of these new distribution methods is the "airdrop", a strategy that involves sending currency or tokens to a predetermined list of wallet addresses or, in some cases, to all holders of a particular blockchain-based asset. Usually a crypto airdrop is a promotional activity performed by blockchain-based startups to help spread awareness about the cryptocurrency or crypto asset project and to get more people trading it. To qualify for the free gift, a recipient may need to hold a minimum quantity of the crypto coin, or token, in the wallet. Alternatively, they may need to perform a certain task, such as posting about the currency on a social media forum, connecting with a particular member of the blockchain project, or writing a blog post.

There are different types of airdrops that can be performed:

- **Standard Airdrop:** participants that are interested in receiving an airdrop simply express their interest. Usually standard airdrops have a set amount of tokens to distribute with a limit on how many tokens each individual may receive.
- **Bounty Airdrop:** occur when users perform certain tasks, that often concern raising awareness of a project, for example by posting on social media.
- **Holder Airdrop:** occur automatically based on who is holding existing tokens and how many tokens they hold.
- **Exclusive Airdrop:** occurs when specific people are individually selected for the airdrop. The difference is they may be selected not based on the amount of tokens they have but based on other elements such as time spent on a project, most money spent on non-token activity, or number of posts in a forum. An exclusive airdrop is an even more centralized way of rewarding those closest to the project and may give airdrops to wallets that may not hold any tokens at all.
- **Raffle Airdrop:** often a project will state the number of airdrops they intend to give and encourage individuals to earn a raffle ticket. This ticket may be earned by holding tokens, earning points, or simply expressing interest.

While airdrops have brought a new powerful democratic distribution method, they also gave the opportunity to a new race of cyber threats to develop and proliferate: "Airdrop scams". The concept behind the airdrop is benign but malicious actors can use it for the own gain exploiting the trust and enthusiasm of the community for their own gain. These scams are designed to use airdrops to distribute tokens or currencies with no value and try to deceive individuals into revealing sensitive information, surrendering funds or downloading malicious software.

Airdrops can function as a tool for project exposure, promote community loyalty and democratizing access to digital assets. Consequently, airdrops have emerged as an effective means of raising awareness, building a dedicated user base, and providing individuals with an introduction to the blockchain space.

## 2.3 Scams

In recent years, blockchain technology has developed quickly, creating new opportunities and gathering attention worldwide.

However, due to its decentralized nature and the user's anonymity, also brought to the invention and development of many fraudulent activities, commonly known as cryptocurrency scams. Scammers use various methods, from the simpler ones to others that are very sophisticated. Social media platforms like Telegram, Twitter, Reddit, and Discord have become primary tools for executing these scams. The interconnected nature of these platforms makes it difficult for investors to avoid falling victim. In the following section we will explore different types of scams that can be performed in the blockchain, with a particular focus on those that exploit airdrop as part of their strategy

### 2.3.1 Phishing

Phishing is a form of social engineering and a scam where attackers deceive people into revealing sensitive information or installing malware such as viruses, worms, adware, or ransomware. Phishing attacks involve malicious actors using social engineering techniques to attain users credentials, install malware on their devices, and obtain their 00 private keys and seed phrases, recovery phrases generated by crypto wallets during setup that enable users to access their wallets if they forget their password or lose their device. A phishing attack usually starts with an attacker sending out a mass email or message to potential victims and it will look like it is sent from a legitimate source. In the message there is almost always a link to a fake website in which the victim inputs their login information. At this point the attacker knows the login information of the victim and uses it to access their account. Usually phishing attacks push the victim to act leveraging on their sense of urgency or fear. In fact most of the messages claim a problem with the user account and that they must log in immediately to fix it.

### 2.3.2 Rug Pulls

A rug pull is a type of exit scam, a type of scams that ends when the fraudster escapes with the funds contributed by participants, that involves a team raising money from the public by selling a token only to quietly shut down the project or suddenly disappear, stealing the raised funds and leaving "investors" with worthless tokens. Rug pulls can be extensively orchestrated, with bad actors leveraging various strategies to lure as many victims as possible. Some scams even use trusted figures to gain trust, while others promise extremely high returns or offer exclusive digital goods. Rug pulls scams are performed using liquidity pools. Briefly, a liquidity pool is a crowdsourced pool of cryptocurrencies or tokens locked in a smart contract that is used to facilitate trades between the assets on a decentralized exchange (DEX). Instead of traditional markets of buyers and sellers, many decentralized finance (DeFi) platforms use automated market makers, which allow digital assets to be traded in an automatic and permissionless manner through the use of liquidity pools. Anatomy of a Rug pull:

- The scammer creates a token and a liquidity pool
- The scammer adds liquidity to the liquidity pool. At this point he is the only owner of tokens in the pool
- The scammer waits until someone swaps valuable tokens with the worthless ones in the liquidity pool
- When someone buys the worthless token swapping it with a valuable one, the scammer removes all the liquidity from the pool.
- The gain of the operation is the valuable tokens that have been swapped minus the gas fees to execute the transaction

The rug pull outlined earlier represents a basic version of the scheme. However, to lure in more investors, attackers often manipulate liquidity pool statistics. One common market manipulation tactic they use is wash trading. In this scenario, the pool creator generates the illusion of activity by artificially inflating the trading volume through repeated buying and selling of tokens. Another strategy attackers employ to attract investors is gradually driving up the token's price over a single day by making incremental purchases, creating the appearance of rising value.

### 2.3.3 Airdrop Scams

The same attributes that make airdrops powerful tools, also make them easy to exploit. There are many deceptive tactics that malicious actor can use to mimic the characteristic of a genuine airdrop. Airdrops are tools that can be used by the scammers to perpetrate any type of scam. They can be used to perform phishing attacks, in fact they often involve fake websites, phishing email or fake social media profiles that promise rewards in exchange for personal information such as private keys of wallet addresses. They can also be used to perform rug pulls, pump and dumps and more types of scams.

The appeal of "free" tokens, coupled with the cloak of authenticity these scams often wear, makes airdrops scams a big threat to the security of blockchain ecosystems. In this thesis, we will delve deeper into the mechanisms and techniques employed by malicious actors to execute airdrop scams, exploring the vulnerabilities they exploit within blockchain systems.



## Chapter 3

# State of the art

In recent years, the phishing detection topic has been extensively studied. Many approaches have been proposed as anti-phishing solutions by many researchers. On the contrary, there are no specific works that focus on airdrop as a technique scammers use to distribute the tokens. In this chapter there is a brief introduction to Machine Learning, a technology that was used extensively during the development of the project, and secondly several works carried out in the field of blockchain scams are presented along with the results these have achieved

### 3.1 Machine Learning

Over the last 30 years, machine learning has become one of the pillars of computing. Machine Learning is a computing technique, developed from the study of pattern recognition and computational learning theory, which through the creation and implementation of algorithms that exploit input data as knowledge, makes systems capable of ‘learning’. The high performance of these algorithms has brought the application of machine learning to various and heterogeneous domains, especially in tasks for which such performance would have been difficult to achieve with classical algorithms. Examples are: spam filters in e-mail, computer vision, natural language processing, etc. The main techniques for training a system are supervised and unsupervised learning. In supervised learning, the data that is used for training is labelled. The label has a specific name: label. The algorithm, in this case, must be able to generalize from examples whose output it knows. Generalization indicates the ability of an algorithm to correctly predict the output of a data, given as input, that it has never seen. Unsupervised classification predicts that the data with which the algorithm is trained does not have an output i.e. a label, associated with it. This type of model is used, for example, to produce clusters of data on the basis of certain statistical properties. The Machine Learning models that have been used during the development of the thesis are supervised.

### 3.2 Related works

The Ethereum blockchain attracts significant financial activity, which in turn draws malicious actors looking to exploit the technology for personal gain. Due to the

unregulated nature of blockchains, illegal market manipulations in traditional stock markets are common in the crypto space. The following describes some of the most recent schemes the research community studied.

**NFT Wash Trading.** The Non-Fungible Token (NFT) market in the Ethereum blockchain experienced growth since 2021. However concerns emerged also for possible frauds realized by trading NFTs. The paper [14] from La Morgia et al. examines the effects of wash trading on the NFT market in Ethereum from the beginning of January 2022, using multiple approaches. Wash trading is a form of market manipulation in which one party repeatedly trades an NFT to inflate its volume artificially. In their paper La Morgia et al. find that wash trading affects 5,6% of the total NFT collections and is responsible for a volume of 3.4 billion US dollars. Wash trading operations are usually performed in a short time window that usually lasts less than 10 days. The key contributions of the paper are: the discovering that approximately 10% of the global NFTs trading volume has been generated through wash trading; finding that most of the colluding traders receive funds from a common account and then send the profit to another common account; many users that participate in a wash trading, also participate in other wash trading activities; wash trading is much more profitable than reselling NFTs legitimately.

**Detecting Phishing Scams on Ethereum based on Transaction Records.** In 2020 Yuan et al. [27] proposed a three steps framework to detect phishing scams on Ethereum by mining Ethereum transactions records. The blockchain system is open and decentralized, so all the transactions records of the Ethereum accounts, including phishing accounts, are publicly available. Yuan et al. mined a big amount of information from the Ethereum transaction records, including the features of the account. From the information of the accounts they tried to extract discriminative features to be able to distinguish a phishing account from a non-phishing account. The framework proposed by the research team to detect phishing scams on Ethereum was realized with the following steps: crawl the labeled phishing accounts and corresponding transaction records from authorized websites, like Etherscan, and according to the records build an Ethereum transaction network; extract features from the transaction network; finally use SVM to distinguish whether the account is a phishing account or not.

**Token Spammers, Rug Pulls, and Sniper Bots.** In the paper by Cernera et al. [3], the research team performs an analysis of the Ethereum and the BNB blockchain from their inception to March 2022. The research team starts by parsing over 3 billion transactions of both blockchains, finding more than 1.3 million tokens and 1 million liquidity pools. Then they reconstructed their lifetime, discovering that around 60% of the tokens have a lifetime shorter than 1 day and those are very active in liquidity pools. Another discovery is that around 1% of the addresses is responsible for creating more than 20% of the tokens. They observed that a large fraction of liquidity pools used to trade the 1-day tokens show a malicious pattern that they called 1-day rug pull. The success rate of the 1-day Rug-pulls is not very high, but they are very easy to perform with a very low cost, so the attackers can cover a series of unsuccessful operations with a single successful one.

**Analysis and Detection of Pump and Dump Cryptocurrency Manipulations.** The pump and dump scheme is a very old fraud scheme, illegal in the stock market. Now it has new vitality in the less regulated market of cryptocurrencies.

Groups of highly coordinated people systematically arrange this scam, usually on Telegram and Discord. La Morgia et al. [15] monitored these groups for a long time and detected around 900 individual pump and dump events. They leveraged their dataset of the verified pump and dumps to build a machine learning model able to detect a pump and dump in 25 seconds from its start.

## Chapter 4

# Datasets

There are several datasets of Ethereum tokens used for analysis tasks but despite this none were designed for airdrop tasks. This chapter introduces and describes the tools we used and the data we collected to create the final dataset that was used to train the Machine Learning model. Several services that were crucial during the development of the project are introduced such as: Etherscan, Phishfort, Metamask, Forta and Webacy.

### 4.1 Tokens dataset

As initial dataset we use the public dataset by Cernera et al. [3]

This dataset contains all Ethereum smart contracts that are ERC-20 compliant until 2023. The dataset contains 389,348 records, each representing a different contract.

Some features of this dataset are already very useful but for the development of the work much more information was needed, more features were needed and also was essential to have some ground truth, namely tokens that performed airdrops that were used to perform something malicious like phishing or some other scam.

### 4.2 Etherscan

Etherscan is a blockchain explorer for the Ethereum network. It provides several helpful functions such as search transactions by hash, search through transactions, blocks, wallet addresses, smart contracts, and other on-chain data. It is one of the most popular Ethereum blockchain explorers and is free to use. It can help understand how people interact with the blockchain, other wallets, and DApps. It can also be used to interact with smart contracts, check gas fees, and search for airdrops. Moreover Etherscan also offers a wide set of APIs and functionalities that can be used to obtain information faster and more convenient formats, i.e. JSON.

#### 4.2.1 Etherscan APIs

The primary goal of using Etherscan and its APIs was to gather comprehensive information about Ethereum tokens and addresses.

The first crucial piece of information to be retrieved was the event logs of each Ethereum token. To obtain the token event logs, the API calls of the Etherscan service were used and specifically the API call that was used is:

1

Using this API, we retrieved the first 1000 event logs for each Ethereum token. This is a limitation of Etherscan, as it can return only up to 1000 logs. The fact that we focus on the initial 1000 events is important because airdrops typically occur early in a token's life cycle.

Initially, we also obtained data on the number of transfers and holders through another API. However, we decided not to use this information because our analysis is limited to the first 1000 events. Knowing the total number of transfers and holders would be counterproductive, as it would include events beyond those covered by our dataset.

### 4.3 Transactions dataset

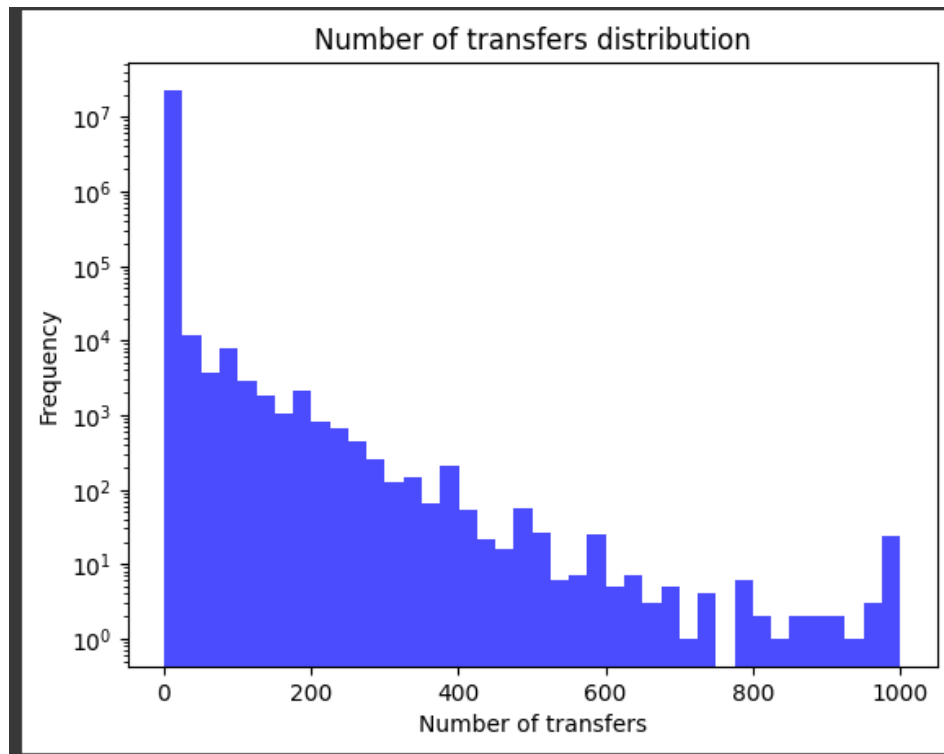
By performing the Etherscan API above for every token in the initial dataset we obtained a JSON, containing the first 1000 event logs, for each token. A crucial figure that we had to check was if the 1000 log cap was a too big limit for the development of the work. As shown in the cdf 4.2 more than the 80% of the tokens in our dataset have less than 1000 events. This means that most tokens in their history have produced less than a thousand events and therefore for most of them we know their entire history.

From the JSONs containing the event logs of the tokens it was possible to create a new dataset containing information about every transaction included in the first 1000 events of the contract. Having this kind of dataset was essential for analyzing the behavior of the transactions and gaining an initial understanding of how airdrop transactions operate. We processed the data and built a more compact representation as follows:

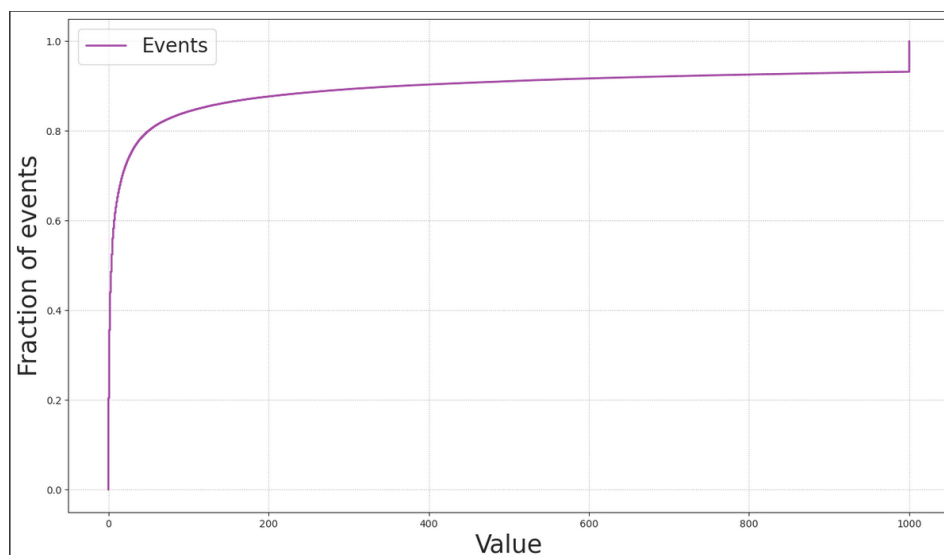
- **Transaction\_hash:** it is the identifier of the transaction within a blockchain. It is unique and distinguishes a transaction from another one.
- **Token\_address:** it is the token contract address, it refers to the address location of the actual token contract that manages the logic for the tokens.
- **n\_transfers:** it is the number of transfers done inside the transaction. This feature was obtained by creating a counter and iterating over the transactions that shared the same transactionHash in the logs, if the transaction had the identified of the event transfer, the hexadecimal string '0xddf252ad1be2c89b69c2b068fc378daa952ba7f163c4a11628f55a4df523b3ef' as first value of the field 'topics', then the transaction was a transfer and could be added to the counter.

---

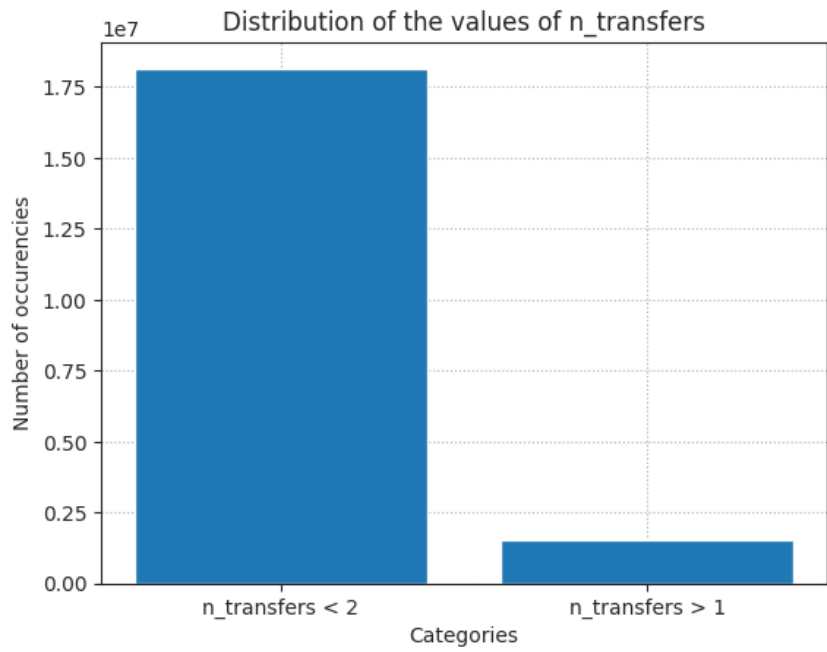
<sup>1</sup>[https://api.etherscan.io/api?module=logs&action=getLogs&fromBlock=block&toBlock=block1&address=address&apikey=api\\_key](https://api.etherscan.io/api?module=logs&action=getLogs&fromBlock=block&toBlock=block1&address=address&apikey=api_key)



**Figure 4.1.** Barplot of transfers distribution in Ethereum tokens transactions



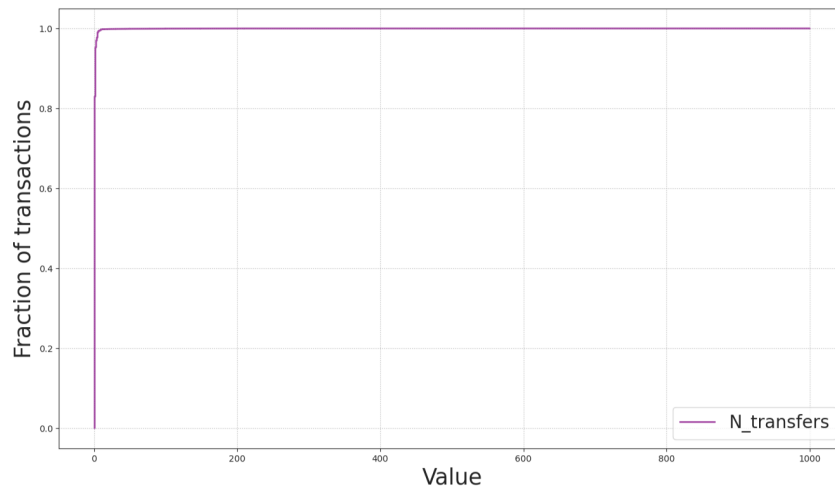
**Figure 4.2.** CDF of events distribution of Ethereum tokens



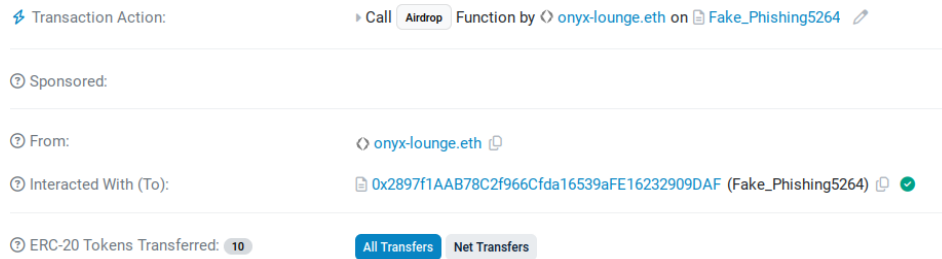
**Figure 4.3.** Barplot showing the distribution of the number of transfer in all token transactions

- **avg\_transfer:** it is the average value transferred in transfers of a same transaction. To compute this feature, it was necessary to obtain the number of tokens exchanged within each transfer. This number is contained in the data field. The data field contains a string in hexadecimal from which it is possible to retrieve the number of tokens that were exchanged in the transfer. This is done by performing the following operation  $\text{int}(\text{data\_value}, 16) * 10^{*(-\text{decimals})}$ . Decimals is the number of decimals used by the token and is information present in the initial dataset. Once the number of tokens exchanged for each transfer of the same transaction has been calculated, calculating the average is very simple.
- **std\_transfers:** it is the standard deviation within the transfers of the same transaction. It is calculated in a very similar way to how it was calculated avg\_transfer. If a transaction contains only one transfer the standard deviation then obviously it is 0.

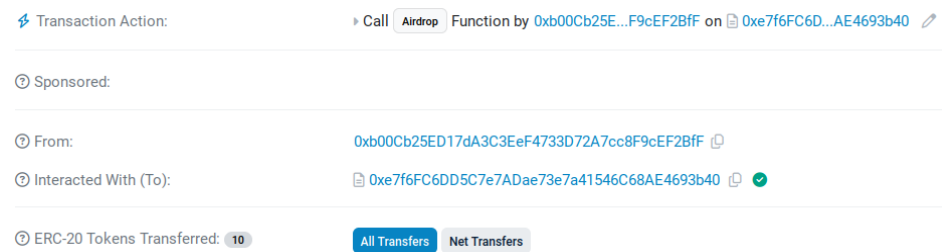
As shown in the CDF 4.4 and in the barplot 4.3 most ERC-20 token transactions have only one or a few transfers. For a transaction to qualify as an airdrop, it needs to contain multiple transfers, not just one. The CDF in picture 4.4 highlights that the number of airdrops is significantly lower compared to all ERC-20 token transactions. In this project we considered an airdrop a transaction that has at least 10 transfers. We chose this threshold after manually checking numerous transactions with around 10 transfers on Etherscan. All of these transactions used functions designed to perform airdrops, as shown in 4.5 and 4.6. It's also worth noting that a function executing an airdrop may have a name different from 'Airdrop'.



**Figure 4.4.** CDF showing the distribution of the number of transfer in all token transactions



**Figure 4.5.** Example of airdrop with 10 transfers



**Figure 4.6.** Example of airdrop with 10 transfers



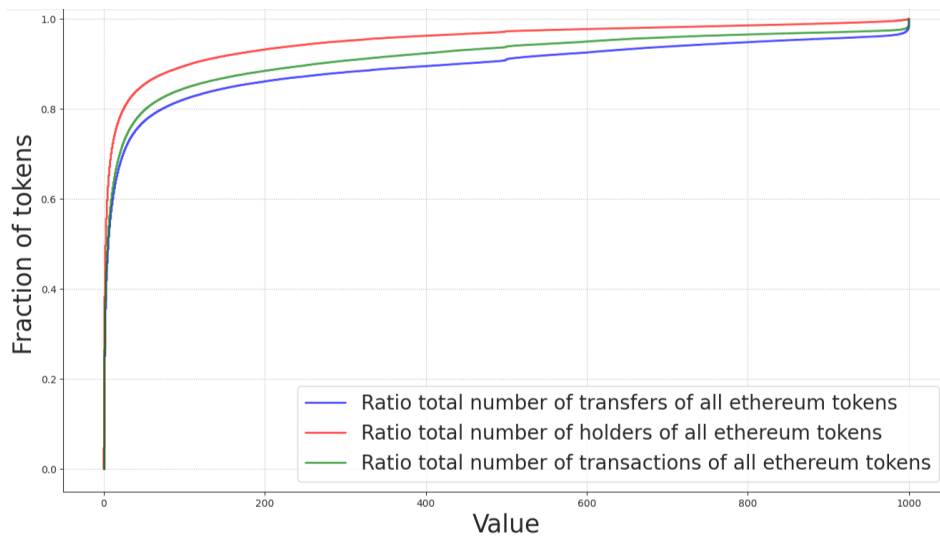
## 4.4 Transfers and holders dataset

Other key features we needed were the total number of transfers, the total number of holders of the token, and the total number of transactions. These values had to be calculated from the logs, not obtained from other APIs, since all our available data pertains to the first 1000 events. As a result, we created a new dataset that includes:

- **Token\_address:** it is the token contract address, it refers to the address location of the actual token contract that manages the logic for the tokens.
- **Total\_transfers:** it is the total number of transfers that were performed with the token. Since the value is retrieved from the logs, the value can't be higher than 1000. To compute it we counted every single event that had in the event type the identifier of the type of event transfer, the hexadecimal string `'0xdddf252ad1be2c89b69c2b068fc378daa952ba7f163c4a11628f55a4df523b3ef'` as first value of the field `'topics'`.
- **Total\_holders:** it is the total number of holders, the addresses that own a certain amount of the token. To compute it we calculated the number of tokens each user exchanged in each transfer in the events and counted how many of these had a balance  $> 0$  tokens at the end of the 1000 events. As for the transfers, a token can not have more than 1000 holders.
- **Transaction\_count:** it is the number of transactions that were performed with the token. As for the `total_holders` and `total_transfers` the transaction count of a token can not be more than 1000.

In a healthy and active token ecosystem, multiple transfers between holders are common as tokens are traded, exchanged, or moved between wallets. For this reason, we presumed that it might be of interest to compute the ratio between holders and transfers and between transactions and transfers. These new metrics were very helpful for us to understand the behavior of Ethereum ERC-20 tokens and find patterns that are common in some malicious airdrop transactions. As an illustration: if a token has a holders/transfers ratio very close to 1, it means that probably the token has not been traded to much. If the number of transfers equals the number of holders, it suggests that each holder has only made a single transfer, which is unusual. Also this pattern could indicate an airdrop scam or fake token distribution, where tokens are distributed to wallets, but there is no real trading or interaction afterward. Each wallet may have only received one token and then never engaged in further transactions, inflating the appearance of token popularity or circulation. In the figure 4.8 we can see that approximately 30% of the tokens have a holders/transfers ratio of 1.

If a token has a transactions/transfers ratio near 0, it can be considered suspicious because it suggests that many transfers are occurring within a single transaction, which could indicate manipulative or unnatural behavior. A low transaction-to-transfer ratio implies that a single transaction is facilitating a large number of token transfers. This often happens in cases of automated or batch transfers, where tokens



**Figure 4.7.** CDF of the distribution of holders, transfer and transaction in ethereum tokens

are distributed to multiple addresses in a single transaction. Airdrop scams use batch transfers to send small amounts of tokens to many addresses, giving the illusion of wide adoption when, in reality, the tokens are not being actively traded or used. In the figure 4.9 we can see that very few tokens have the ratio near 0, while most of them (65% approximately) have the ratio equal to 1.

## 4.5 Differences between the creation and the first event and between the first and last event

Other important important features that we decided to have were: the differences between the first event of the token and its creation and between the last event and the first event. These information was obtained from the logs of the events of the tokens. The resulting dataset that we created was composed in this way:

- **Token\_address**
- **Difference\_creation\_first\_event:** it is the difference between the timestamp of the first event of the token and the timestamp of the creation event of the token. The information of the timestamp where retrieved from the field "timestamp" in the logs and they express the block in which the event took place. So the differences are expressed in terms of blocks.
- **Difference\_first\_last\_event:** it is the difference between the timestamp of the last event of the token and its first event. Exactly as for the difference between the first event and the creation, the difference is expressed in blocks.

If the time between the creation of a token and its distribution is extremely short, this could mean that the token is following a pre-planned or automated behavior. In the context of airdrop scams, the frauders can create the token and immediately start distributing it. The immediate activity after the creation suggests that the



Figure 4.8. CDF of the distribution of the ratio between holders and transfers

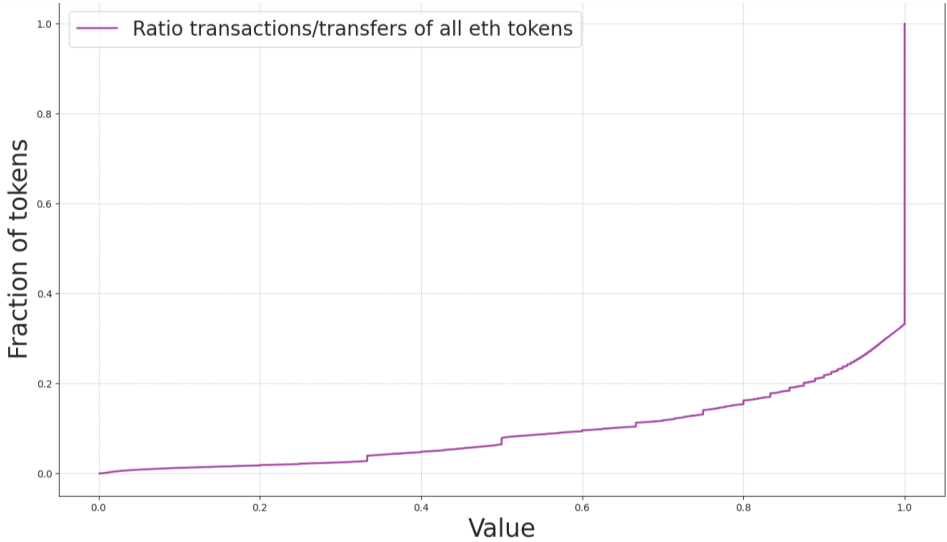
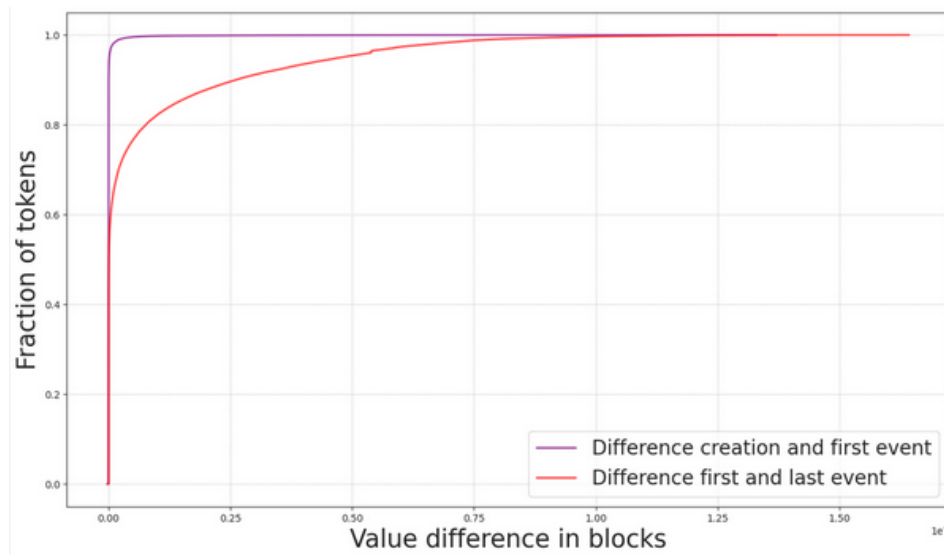


Figure 4.9. CDF of the distribution of the ratio between transaction and transfers



**Figure 4.10.** CDF of the distribution of the differences between the first event and the creation and between the last event and the first event

token did not go through normal growth, which typically takes time. We also define the lifespan of a token as the difference between the first and last event. If the lifespan is very short, this suggest that there has been a very intense activity in a very narrow window, that is also suspicious. Scam tokens usually are exchanged heavily after creation and then they become inactive. A scammer can create a token, immediately make airdrops to different addresses with malicious intents and then stop exchanging the token. Legitimate tokens, on the other hand, tend to have ongoing activity over a longer period.

Both these behaviors suggest that the token's activity was not organic, but rather a part of a suspicious or fraudulent scheme.

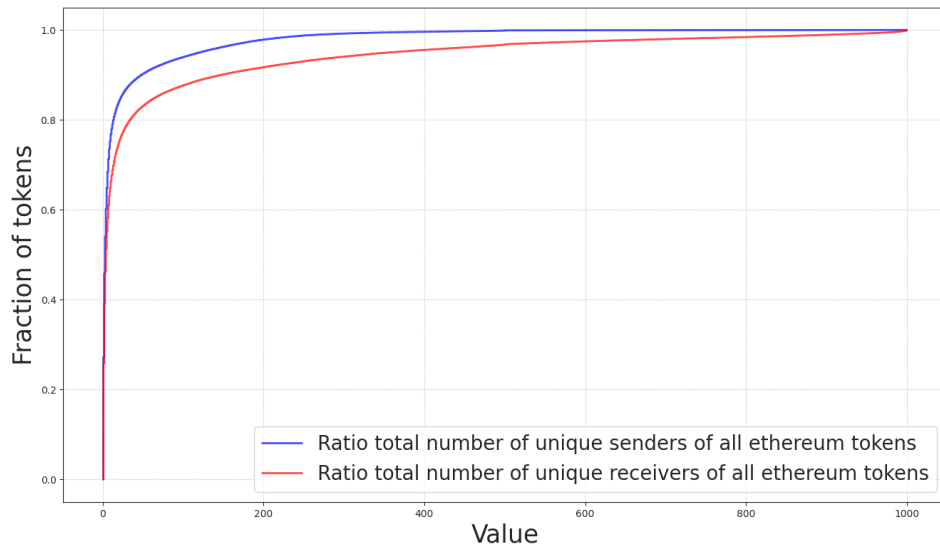
Figure 4.10 shows that almost the totality of the tokens (around 95%) have the first event and the creation event difference at 0, this means that they are created and distributed in the same block. The CDF also shows that around 60% of the tokens perform the last event in the same block where they performed the first event.

## 4.6 Senders and receivers

The last type of information we thought might be useful to obtain is the number of senders and receivers. Having these values available makes it possible to detect repetitive patterns in transactions. In a healthy environment, the number of senders and receivers should be similar.

This dataset is composed in the following way:

- **Token\_\_address**
- **Number\_\_of\_\_senders:** represents the number of unique addresses that sent



**Figure 4.11.** CDF of the distribution of senders and receivers

at least one time the token to some other address. In the log of each event, if the event is a transfer, the second value of the field "topics" is the address of the sender.

- **Number\_of\_receivers:** represents the number of unique addresses that received at least one time the token from some other address. In the log of each event, if the event is a transfer, the third value of the field "topics" is the address of the receiver.

If a token has a high number of unique receivers but a very low number of unique senders, it can probably be an airdrop or automated distribution.

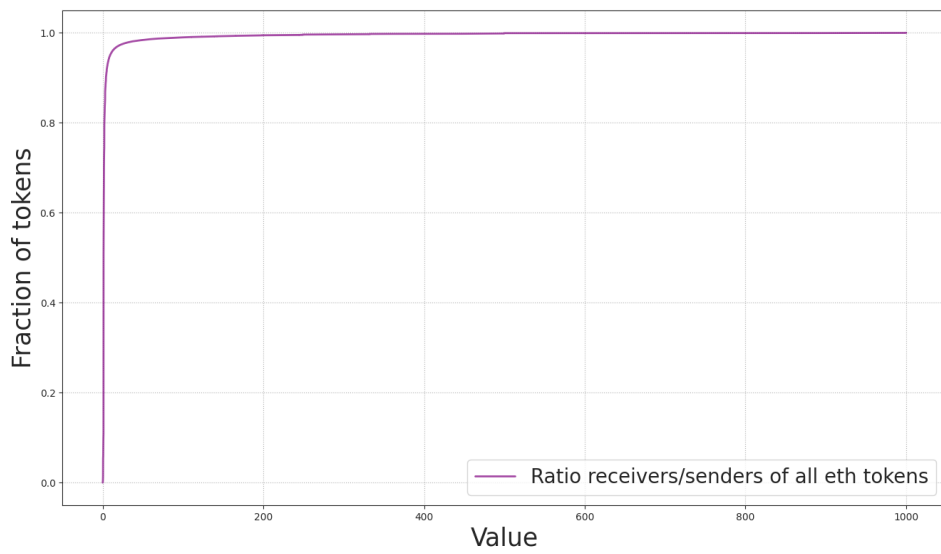
If a token has a high number of senders but a low number of receivers may indicate a pump and dump scheme. A small number of accounts could be collecting tokens from many different sellers, which might happen when the price of the token has been artificially inflated.

Large imbalances between the number of unique senders and receivers can indicate suspicious behavior.

In figures 4.11 and 4.12 we can see the CDFs of the distribution of the senders and receivers and the ratio between them.

## 4.7 Ground truths

Ground truth data is a critical component for training machine learning models. It serves as a reference for verifying predictions. Since the task we faced is new, there is not much public data and no ground truth for our classification task either. Building a labelled dataset large enough to train a Machine Learning model was a very complicated task, mainly due to the fact that there are no unique sources. Therefore, we thought of finding ground truth using different tools, which are presented in this section. This section will outline the methodology used to acquire ground truths.



**Figure 4.12.** CDF of the distribution of the ratio between receivers and senders

We marked with label 1 a token that performed at least one airdrop and performed some malicious event.

#### 4.7.1 Metamask eth-phishing-detect

MetaMask [18] is a software wallet made specifically for tokens and dapps on the Ethereum blockchain. It is available as a browser extension and mobile application, which can be used to buy, sell and swap coins and access decentralized applications. Staking is available directly from the wallet with third-party integration, and non-fungible token, or NFT, marketplaces are available on the mobile app and via the browser extension.

Metamask provides a dataset called eth-phishing-detect [17]. It is a list containing malicious domains targeting Web-3 users, it is open and accessible to anyone. With this list an initial ground truth labelling malicious tokens was created. The matches (contracts that were in both datasets) between our initial dataset and the list eth-phishing-detect were 10 tokens but out of those only 2 had at least a transaction that performed an airdrop.

#### 4.7.2 Phishfort

Phishfort [22] is brand reputation and phishing prevention platform. Similarly as Metamask Phishing detect Phishfort offers a list [21] of known malicious domains and tokens and this was also used to have an initial ground truth on domains and tokens that are malicious for sure.

The matches between the Phishfort list and the initial dataset were 800, but from those only 8 had at least one transaction that performed an airdrop.

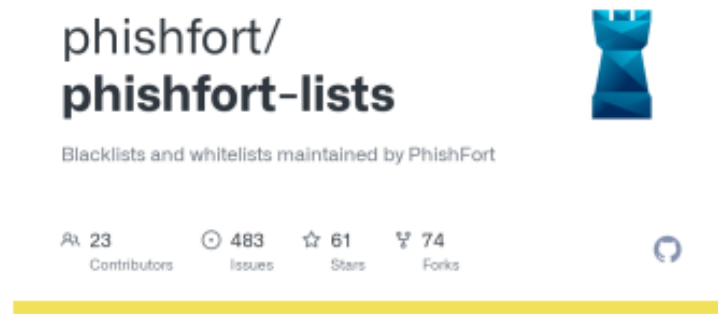


Figure 4.13. Github repository of Phishfort

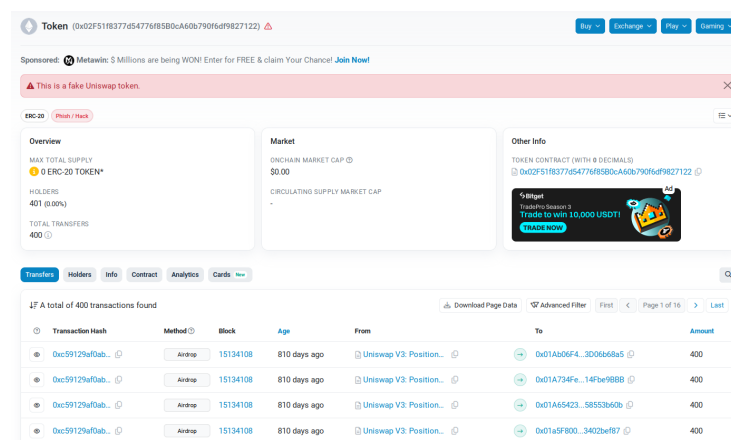


Figure 4.14. Example of ERC-20 token that has the phish/hack label on Etherscan

### 4.7.3 Forta network

Forta [9] is one of the largest network of security intel in Web3. The decentralized Forta Network leverages machine learning and a community of security researchers to detect exploits, scams and other threats. It is a powerful Web3 protocol that generates actionable and insightful threat intelligence.

Forta provides datasets of suspicious Web3 activity on different chains. In particular there are datasets [10] in which are present some ethereum addresses that have the label "phish/hack" on Etherscan. Figure 4.14 shows one of the ERC-20 tokens that has the label on Etherscan. That token is present in the dataset from Forta.

The dataset contains around 8000 records but many of them are users addresses, not contracts. Between the contracts subset, few of the records are ERC-20 tokens and from those, very few of them had a least a transaction that performed an airdrop. The number of ground truths that we were able to obtain from the Forta dataset was around 50 malicious tokens.

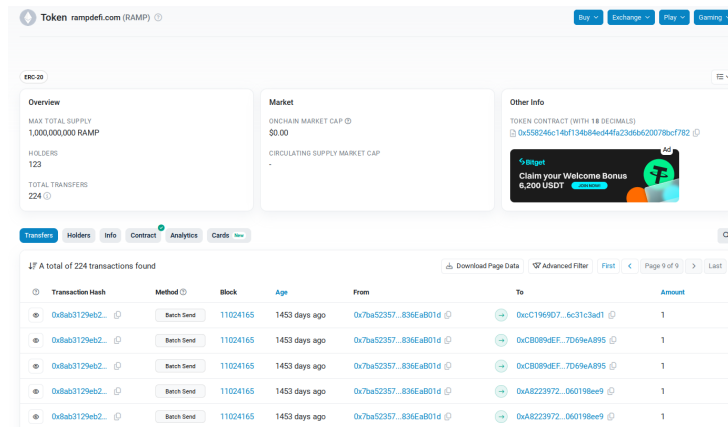


Figure 4.15. Example of one day token that performed also an airdrop

#### 4.7.4 One day tokens

Another dataset that we used to gather more ground truth is the "one day token" dataset. It is a public dataset by Cernera et al. [3] and was created starting from the public dataset that we use as initial dataset. The "one day token" dataset contains 82,542 tokens that had a lifetime equal or shorter than 1 day. A token with a very short life span is suspicious. As proven in the research study by Cernera et al. [3] all of these token are used to perform actions like Rug-pulls and Spam.

Out of all the tokens in the "one day token" dataset 1758 of them had at least a transaction that performed an airdrop. All those tokens can be used as ground truth labeling malicious tokens. In figure 4.15 there is an example of the tokens labeled as malicious, it is a one day token that also performed an airdrop.

#### 4.7.5 Webacy

Webacy [25] is an application-layer safety platform that helps wallets and applications protect their users against scams, hacks and mistakes across the blockchain. Webacy gives information through Etherscan about the safety of the token that is being analyzed. Webacy offers some APIs that allow to perform real-time queries to their risk decisioning engine and retrieve risk information about transactions, addresses or smart contract addresses. We used the service by calling the following API:

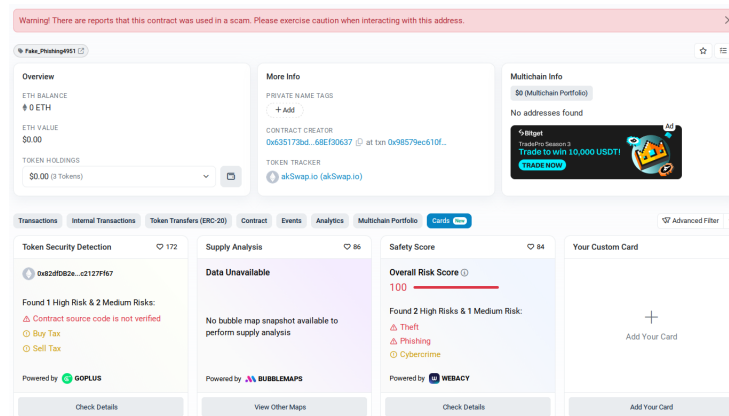
2

The API call returns a JSON containing much information about the contract, including a field called "risk" that is the overall risk the service derived from scanning all known information about the token. If the risk level is high is very likely that the token is malicious.

The risk information of the assets gathered from webacy turned out to be very useful for the development of the project, since it could be used as ground truth labeling a token as malicious if the risk of the token is high and non-malicious otherwise.

<sup>2</sup>[https://dapp.webacy.com/api/risk/contracts/contract\\_address?chain=eth](https://dapp.webacy.com/api/risk/contracts/contract_address?chain=eth)





**Figure 4.16.** Example of ERC-20 token to which Webacy gives high risk score

In figure 4.16 is shown a token on Etherscan that has a very high level of risk indicated by the service Webacy

#### 4.7.6 Non malicious airdrops


All the ground truth that was found previously only labeled malicious behavior, we did not have the same quantity of labels for tokens that performed non malicious airdrops. The objective was to create a dataset to train a machine learning model and, in order to do so, the dataset had to be balanced, with a similar number of malicious and non-malicious records.

To find the "benign" airdrop tokens we searched online for airdrop campaigns realized by important companies to advertise a real project. Figures 4.17 and 4.18 are two examples of advertisement of airdrop campaigns of very popular and legit tokens.

#### 4.7.7 Manual labeling

From the tools described before we were able to find some ground truth and add them in the dataset we used to train the ML model that predicts if a token is malicious or not. However the quantity of ground truth found just with automated tools or public datasets was not enough. The training dataset needed to be much bigger in order to train the dataset adequately. So the next step was to find manually the ground truth and manually labeling some tokens that did airdrops. The first step was to take a big number of tokens that probably did an airdrop transaction, a transaction with 10 or more transfers inside, and check that the transaction was an airdrop. From all the many tokens that we manually checked we noticed that 10 transfers in a transaction is a sufficient threshold to prove that the transaction is an airdrop. After checking if the token performed an airdrop, the next step was to understand if the token had some malicious behavior or not. To do so we searched information online for information on tokens that could clearly state whether the token was malicious or not. These type of information can usually be found in forums or social networks. In figure 4.19 is shown an example of the tokens we found that were reported malicious in online forums.

# Uniswap



9\*

Airdrop Link: [MORE INFORMATION](#)

Total value: 600,000,000 UNI + 5,000,000 USD

Platform: eth

Uniswap, a fully decentralized protocol for automated liquidity provision on Ethereum, released its governance token UNI.

**Uniswap is conducting multiple airdrops to historical users of Uniswap and also to historical users of Genie.**

**Airdrop 1: Historical Genie users (up to 1,000 USD)**

Uniswap is airdropping a total of 5,000,000 USD to certain historical Genie users based on a snapshot taken on April 15th, 2022 at 00:00 UTC. Genie was the first NFT marketplace aggregator, which was acquired by Uniswap back in June.

Users that completed more than one transaction before the snapshot can claim 300 USD and users that held the Genie Genesis NFT as of the snapshot can claim 1,000 USD.

**Airdrop 2: Historical Uniswap users (up to 1,000 UNI)**

Uniswap is also airdropping 60% of the UNI genesis supply to Uniswap community members, a quarter of which (15% of total supply) can be claimed by historical users, liquidity providers, and SOCKS redeemers based on a Snapshot on September 1, 2020 12:00 am UTC.

400 UNI are claimable by each address that has ever called the Uniswap v1 or v2 contracts. This includes ~12,000 addresses that have only ever submitted failed transactions.

**49 million UNI** are claimable by historical liquidity providers. The formula accounts for LP liquidity on a per-second basis since the deployment of Uniswap v1, ensuring that rewards are weighted towards LPs that provided liquidity when total liquidity was low.

1000 UNI are claimable by each address that has either redeemed SOCKS tokens for physical socks or owned at least one SOCKS token at the snapshot date.


**Step-by-Step Guide:**

**Airdrop 1: Historical Genie users (up to 1,000 USD)**

1. Visit the [Uniswap website](#).
2. Connect your wallet.
3. Click the wallet dropdown in the top right.
4. If you're eligible then you will be able to claim free USD.
5. Users that completed more than one transaction before the snapshot can claim 300 USD and users that held the Genie Genesis NFT as of the snapshot can claim 1,000 USD.
6. The snapshot was taken on April 15th, 2022 at 00:00 UTC.
7. For more information regarding the airdrop, see this [article](#).

**Figure 4.17.** Uniswap airdrop advertise campaign

# Ethereum Name Service



Ethereum Name Service is a distributed, open, and extensible naming system based on the Ethereum blockchain. ENS's job is to map human-readable names like 'alice.eth' to machine-readable identifiers such as Ethereum addresses, other cryptocurrency addresses, content hashes, and metadata.

Ethereum Name Service is airdropping 25% of the total supply to 'ETH' domain holders. The snapshot was taken on October 31st, 2021 and eligible users have until May 4th, 2022 to claim the tokens.

**Step-by-Step Guide:**

1. Visit the [Ethereum Name Service airdrop claim page](#).
2. Connect your ETH wallet.
3. If you're eligible, then you will be able to claim free ENS tokens.
4. A total of 25% of the total supply has been allocated to eligible users.
5. The snapshot was taken on October 31st, 2021.
6. Users who've or have been a registrant of a 'ETH' second-level domain by the snapshot date are eligible for the airdrop.
7. The individual allocation will be based on the number of days the account owned at least one ENS name and the days until the expiration of the last name on the account.
8. There's also a 2x multiplier to accounts that has a Primary ENS Name set.
9. Eligible users have until May 4th, 2022 to claim the tokens.
10. For more information regarding the airdrop, see this [article](#).

Don't forget to follow us on [Twitter](#), [Telegram](#), & [Facebook](#) and [subscribe our newsletter](#) to receive new airdrops!

**Figure 4.18.** Ethereum name service airdrop campaign

**Warning:** One or more bitcoin.org users have reported that they strongly believe that the creator of this topic is a scammer. (Login to see the detailed trust ratings.) While the bitcoin.org administration does not verify such claims, you should proceed with extreme caution.

---

Pages 1/3  
Author  
grayscaledeveloper (OP)  
Donor  
Activity 1  
View 1

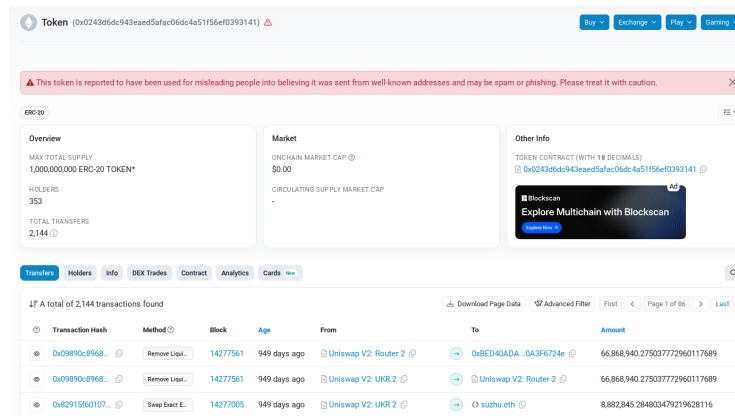
Topic: graycash (graycash) - a complete crypto ecosystem (read all posts)  
☐ graycash (graycash) - a complete crypto ecosystem  
May 10, 2019, 10:15 AM UTC  
[https://graycash.org/en/home/about/hoggen/joining.php](#)  
**GrayCash**  
Graycash (graycash) is the latest blockchain platform designed to provide a solution for a complete crypto ecosystem. It was developed by using ERC-20 infrastructure for easier market adoption and compatibility.  
Official website  
[@graycash](#)

---

- Current name: GrayCash
- Token name: GDCN Tokens
- Token symbol: GDCN
- Token type: (Ethereum ETC2)
- Token supply: 100M
- Total tokens issued: 10,000,000 GDCN Tokens
- Contract Address: [0x1c0f9e9dca264841cf7c222440179401914](#)
- Block Explorer: [https://explorer.graycash.org](#)
- ICO Website: /ICO
- Mining: Together with Ethereum Blockchain
- Minimum trade price: 0.01 Bitcoin per GrayCash token = 1 GDCN = 1000 USD
- Minimum distribution: 0.01 Public sale of 10 tokens created 15% Company's reserve

DO NOT MISS THIS EXCLUSIVE ICO OPPORTUNITY TO PARTICIPATE IN **GrayCash** NETWORK NOW!

**Figure 4.19.** Example of token that is reported to be malicious in online forums



**Figure 4.20.** Example of token that is reported to be a phishing scam on Etherscan

Furthermore, since there is no information on many tokens in online forums, we checked manually tokens on Etherscan. From the etherscan view we were able to have a better view of the tokens, the transactions of the tokens, the name of the methods called and information from other security services like Webacy. During the manual labeling we considered malicious the tokens that fall into at least one of the following categories:

- Tokens that have comment on Etherscan reporting that the token is with high probability a phishing scam, example in 4.20
- Tokens that have comment on Etherscan reporting that the token has been spammed to a large number of addresses, example in 4.21
- Tokens that have the phish/hack label on Etherscan, example in 4.22
- Tokens that have other types of comments on Etherscan, as in figure 4.23
- Tokens that are fake copies of other tokens, example in figure 4.24
- Tokens for which the owner can change the balance, as in figure 4.25
- Tokens that clearly performed a Rug pull. The anatomy of a rug pull is the following: open a liquidity pool on the token, wait for somebody to exchange tokens, close the liquidity pool.
- Tokens that are classified as fake tokens in the Webacy card on Etherscan, as in figure 4.26

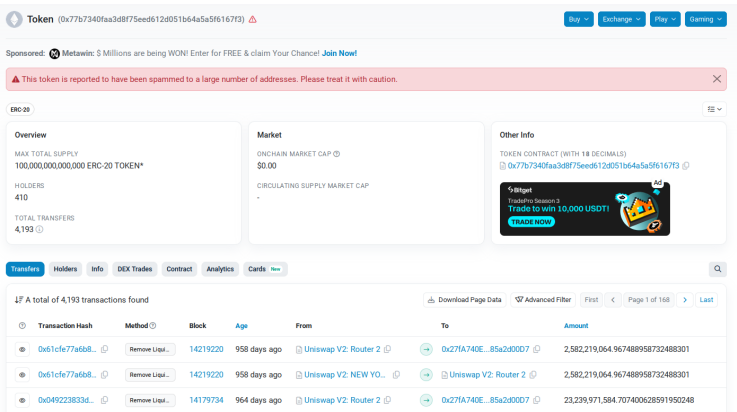


Figure 4.21. Example of token that is reported to be spam on Etherscan

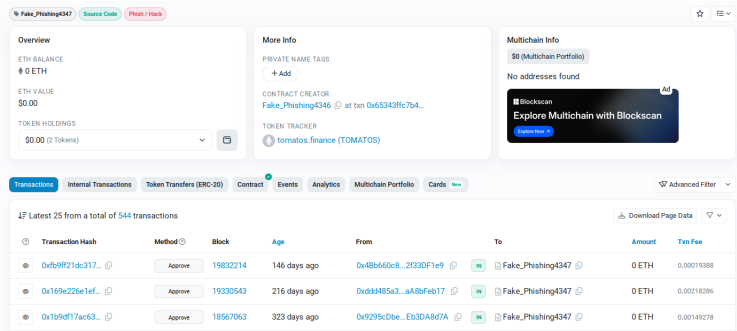


Figure 4.22. Example of token that has the phish/hack label on Etherscan

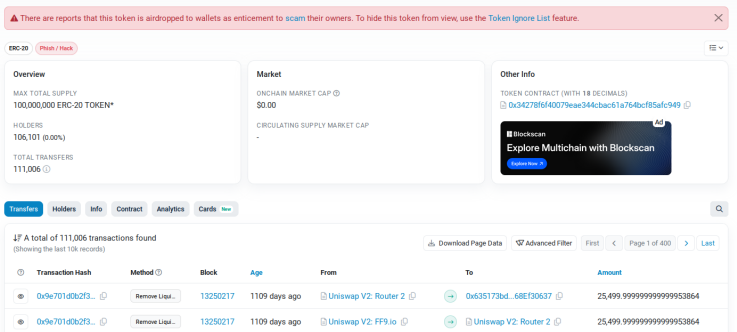


Figure 4.23. Example of token that performed an airdrop scam

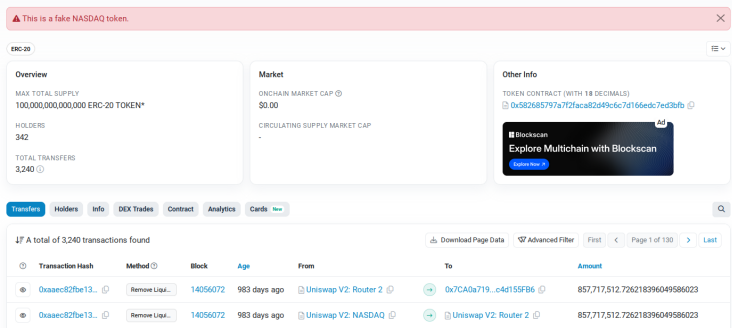


Figure 4.24. Example of token that is a fake token

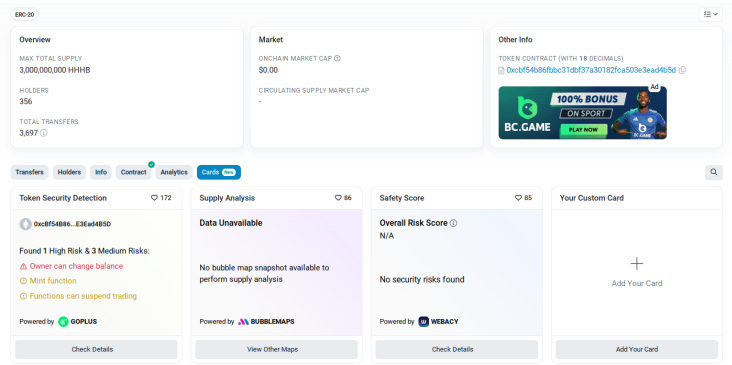


Figure 4.25. Example of token for which the owner can change the balance

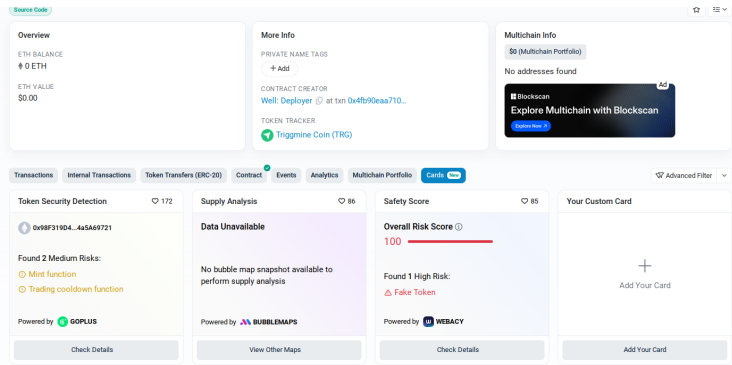


Figure 4.26. Example of token classified as fake token

## Chapter 5

# Methodology

This chapter describes the technical work that has been done to build the proposed machine learning model. The first part describes the features that have been implemented in the final dataset used to train the Machine Learning models. The second part concerns the different ML models that were used during the development of the project, their description, and the rationale behind their choice.

### 5.1 Features

A feature is an individual measurable property or characteristic of a phenomenon. It is an element whose value, or representation of it, is a good discriminator for the separation between two or more different classes. Choosing the right features is a crucial element of effective algorithms in pattern recognition, classification or regression.

Upon testing and engineering different features, we select the following:

- **Difference\_creation\_first\_event**
- **Difference\_first\_last\_transfer**
- **N\_transfers:** it is the number of transfers realized in the transaction with most transfers of the token
- **Avg\_transfer:** it is the average of tokens exchanged in the transactions with most transfers of the token
- **Std\_transfer:** it is the standard deviation of tokens exchanged in the transactions with most transfers of the token
- **Total\_transfers**
- **Total\_holders**
- **Transaction\_count**
- **Transactions\_transfers**
- **Transfers\_holders**

- **Number\_of\_senders**
- **Number\_of\_receivers**
- **N\_transfers/\_senders**
- **N\_transfers/\_receivers**
- **Name:** it is the name of the token
- **Total\_supply:** is the total number of tokens that has been created or mined.

### 5.1.1 Chargrams

All the features of the training dataset are numerical ,but for one, "Name", that is categorical. Machine learning models are usually based on mathematical operations, that, of course, require numbers. So a feature that is not numerical can not be given as input to the algorithm. When working with text it is necessary to come up with a strategy to convert strings to numbers before feeding it to the model. The conversion from string to number is called "vectorization". There are different ways to vectorize a string such as:

- One-hot-encoding: converts categorical data in binary vectors.
- Label encoding: assigns an identifier to each string
- Embedding: converts strings into dense vector representations
- N-grams: n-grams are contiguous sequences of 'n' items. These items can be characters, words, or even syllables, depending on the granularity desired. The value of 'n' determines the order of the N-gram

We decided to opt for N-grams, specifically Chargrams. A n-chargram is a sequence of N adjacent characters from a particular source of text. In our case it is very useful because can help the model to find patterns in the names of the tokens of the dataset. For example many malicious tokens could share a n-gram with other tokens. Many malicious tokens have similar patterns in their name, this means that they can have the same characters grams. For instance token can have a have a domain in their name, as visible in figure 5.1, and this means that those tokens have similar patterns in their name (they have a point character). They can have also other patterns in common, like special characters, an identical part of the name or entire words that are present in different malicious tokens. Figures 5.2and 5.4 show two different fake tokens that have the same pattern in their name,in fact both tokens have the word "Metaverse" in their name. In figures 5.3 and 5.5 there is the token tracker view of the tokens, where both tokens are reported as malicious.

**Char n-gram that we used:**

- uni-gram
- bi-gram
- tri-gram
- four-gram

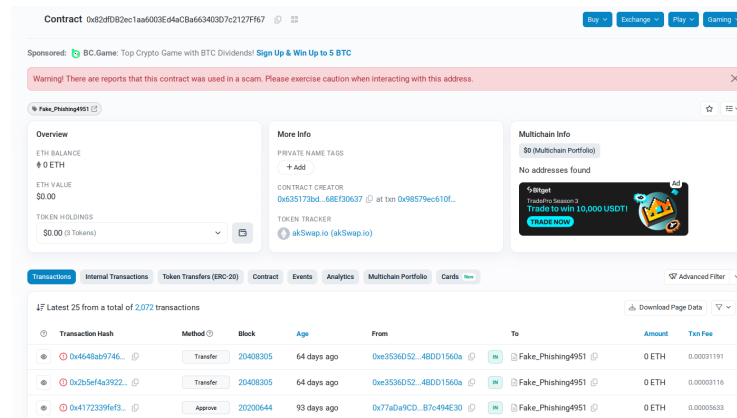


Figure 5.1. Ethereum token with a domain in the name

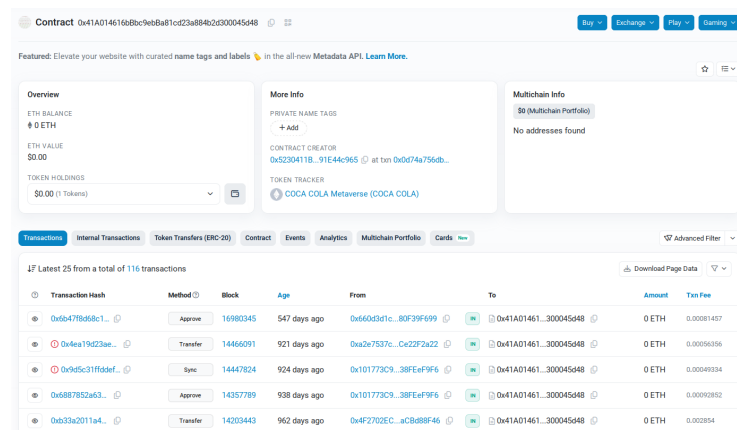


Figure 5.2. Fake coca cola token

- five-gram
- six-gram

### 5.1.2 Feature scaling

Normalization [23] is a technique often applied as part of data preparation for Machine Learning. The goal of normalization is to change the values of numeric columns in the dataset to a common scale, without distorting differences in the ranges of values. For machine learning, every dataset does not require normalization. It is required only when features have different ranges. The ranges in the features of our dataset vary widely so normalization is necessary. Feature scaling is a technique used to normalize the range of features in a dataset, transforming the data to make it more optimized for modeling. We used the Scikit-Learn libraries to perform this task.



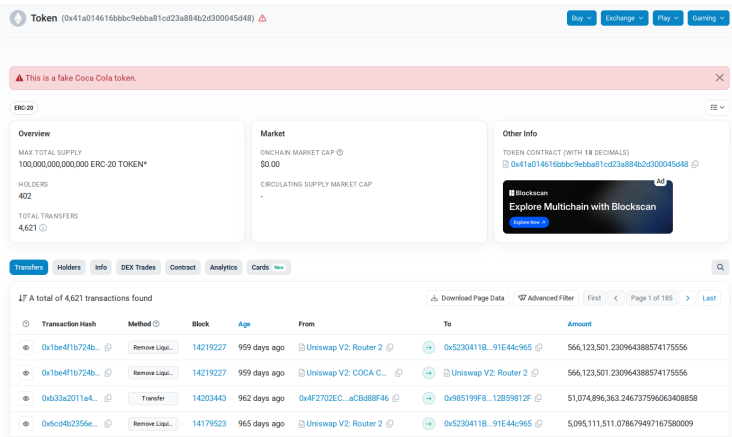


Figure 5.3. Token tracker view of the fake coca cola token

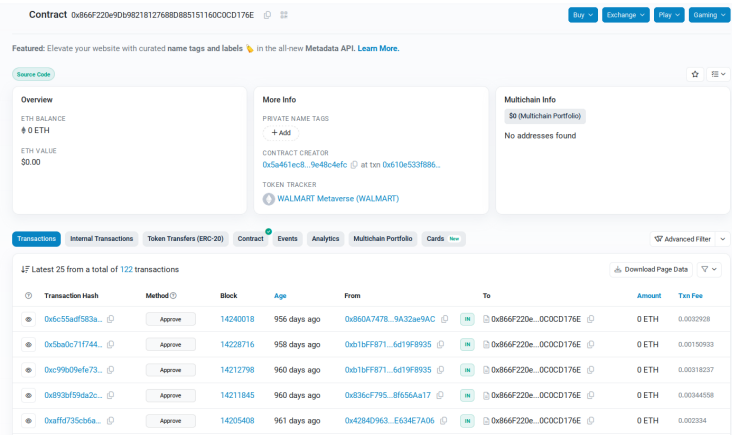


Figure 5.4. Fake Walmart token

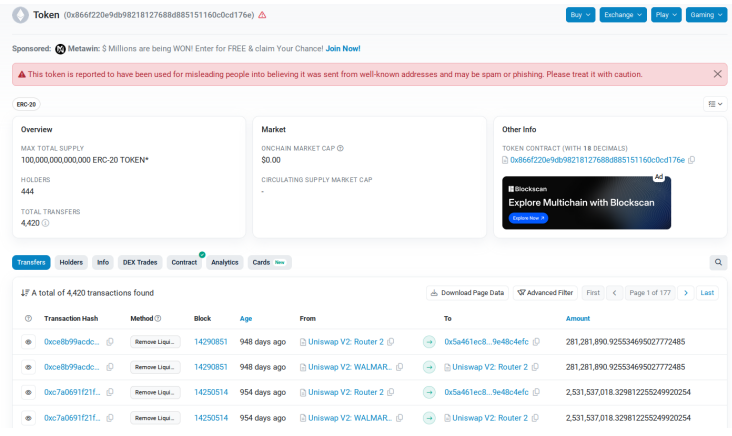


Figure 5.5. Token tracker view of the fake Walmart token

## 5.2 Models

We used different models during the development of the project but we did not use any Deep Learning algorithm. Indeed Deep Learning are very powerful but need a large amount of data to be trained in an effective way, avoid overfitting and learn complex representations of data. Traditional Machine Learning models are less powerful and need to optimize less parameters, so they can perform better even with small dataset.

### 5.2.1 Random Forest

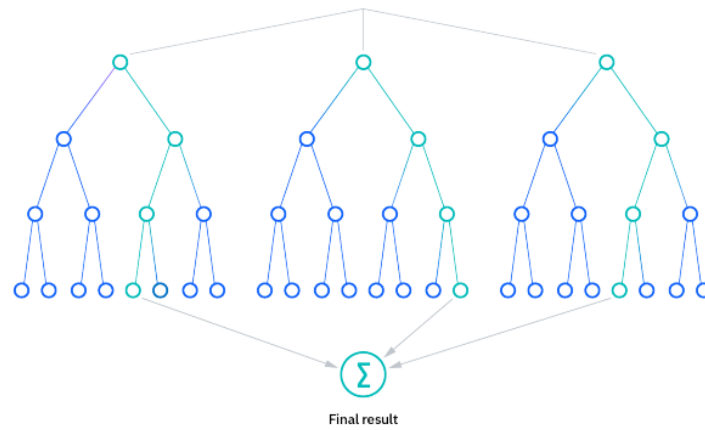
The first model we used to classify tokens is the Random Forest. It is a machine learning algorithm that combines the output of multiple decision trees to reach a result. A decision tree is a supervised learning algorithm that has a hierarchical tree structure that consists of a root node, branches, internal nodes and leaf nodes. The decision tree starts with decision nodes that act as means to split the data according to specific characteristics. Each division brings the algorithm closer to final decision, which is denoted by the leaf node.

Random forest [1] is an ensemble learning method, it is made up of a set of classifiers, decision trees, and their predictions are joined to identify the most frequent result or the average of the estimates, depending on the type of problem. The most famous ensemble methods are:

- **Bagging:** is an ensemble method that involves training multiple models independently on random subsets of the data and joining their predictions through voting or averaging. Each model is trained on a random subset of the data sampled with replacement, meaning that the individual data points can be chosen more than once. This random subset is known as a bootstrap sample. By training models on different bootstraps, bagging reduces the variance of the individual models. It also avoids overfitting by exposing the constituent models to different parts of the dataset.
- **Boosting:** it is similar to bagging but differently from bagging, with boosting the models are trained sequentially, rather than independently, with each model learning from the errors of the previous one.

The random forest is an extension of the bagging as it also utilizes feature randomness to generate a random subset of feature, which ensures low correlation among decision trees. The random forest algorithm is made up of a collection of decision trees, and each tree in the ensemble is comprised of bootstrap sample. Of that training sample, one-third is set aside as test data, known as the out-of-bag (oob) sample. Another instance of randomness is then injected through feature bagging, adding more diversity to the dataset and reducing the correlation among decision trees. Depending on the type of problem, the determination of the prediction will vary. For a regression task the prediction is the average of the decision trees, for a classification task it is the most predicted class by the decision trees. Finally, the oob sample is then used for cross-validation, finalizing that prediction.

The main reasons why we chose Random forest as the initial model were the robustness to overfitting and the ability to handle imbalanced datasets. Among the



**Figure 5.6.** Structure of Random forest

models we test, we prefer the model that is robust to overfitting and that better perform in case of imbalance in the dataset and random forests have a good behavior in both situations.

In addition random forests can provide insights about the importance of each feature in the classification task, helping us understand which features are most important in distinguishing malicious from non-malicious airdrops.

### 5.2.2 SVM

SVMs [24] are supervised learning models with associated algorithms, which use labelled data for classification. After training the SVM through a set of examples, each of which is labelled with the correct label, it is able to non-probabilistically associate a label with each new given example. probabilistically a label to each new example given. For classification, the SVM model represents data as points in a space, and for each class, it associates a label with it. New examples are mapped within this space and associated with the respective label. Formally, the SVM constructs a hyperplane or a set of them, in an infinite or many-dimensional space that is used for classification, regression or other tasks such as the identification of outliers. A good separation is achieved by the hyperplane that has the largest distance widest between the nearest training points of each class. The higher the margin between the points and the hyperplane, the smaller the generalisation error of the classifier.

SVM is one of the most widely used models for NLP tasks, so we chose it because it should perform very well in pattern recognition in ‘Name’ features. Furthermore, SVMs are effective in cases where there is a high dimensional space, where there are many features and in cases where the dataset is small.

### 5.2.3 Gradient boosting

Gradient boosting [11] is a popular boosting algorithm in Machine Learning used for classification and regression tasks. It combines several weak learners into strong learners, in which each new model is trained to minimize the loss function such as

mean squared error or cross-entropy of the previous model using gradient descent, a method for unconstrained mathematical optimization. In each iteration, the algorithm computes the gradient of the loss function with respect to the predictions of the current ensemble and then trains a new weak model to minimize this gradient. The predictions of the new model are then added to the ensemble, and the process is repeated until a stopping criterion is met. Each predictor is trained using the residual errors of the predecessor as labels.

Gradient boosting is very efficient on binary classification, for its robustness and interpretability it proved to be an excellent choice.

## Chapter 6

# Test & Results

In this chapter, we show the tests and report the performance of 3 models we implemented for classifying tokens into malicious or non-malicious. Before training the models with the dataset, we ran tests to tune the hyperparameters so that each model could offer the best possible performance. Specifically, we use the `GridSearchCv` library function from `scikit-learn`. `GridSearch` takes different values for the hyperparameters and tests all combinations using cross-validation. Cross-validation is a technique that evaluates the performance of a machine learning model by partitioning the data into multiple subsets. It involves training the model on some of these subsets and testing it on the remaining data, rotating the subsets to ensure every part of the data is used for both training and testing. This approach helps in assessing how well the model generalizes to unseen data and reduces the risk of overfitting, especially when working with limited datasets. After using this function, we select the combination of hyperparameters that yields the best performance.

We randomly divided the dataset into two parts: training set (80% of the dataset) and test set (20% of the dataset).

### 6.1 Random Forest

After running `GridSearch`, we trained the model using the following hyperparameters:

- `N_estimators = 300`; represents the maximum number of trees in the forest
- `Max_depth = None`; represents the maximum depth of each tree
- `Min_samples_split = 5`; it is the minimum number of samples needed to split a node
- `Min_sample_leaf = 1`; it is the minimum number of samples needed to be a leaf node

Figure 6.1 shows the evaluation metrics of the model. The confusion matrix is shown in 6.1. As we can see, our model has a very high values in the evaluation metrics, proving to be a good classifier for this task. For our model, it is important that the recall value is as high as possible. This metric indicates the ability of a model to correctly identify positives among all true positives in the data. The model obtains a recall value of 90%.

```

Accuracy: 0.8888888888888888
Classification Report:
              precision    recall  f1-score   support

     0       0.86      0.91      0.89        77
     1       0.91      0.87      0.89        85

 accuracy          0.89        162
 macro avg         0.89        162
 weighted avg      0.89        162

```

**Figure 6.1.** Random Forest metrics**Table 6.1.** Random Forest confusion matrix

		Predictions	
		0	1
Values	0	70	7
	1	11	74

## 6.2 SVM

After running GridSearch, we trained the model with the following hyperparameters:

- $C = 0.1$ ; It is the regularization parameter, it tells the algorithm how much to avoid misclassifying each training sample. It is inversely proportional to the margin size, the larger the value of  $C$ , the smaller the margin and vice versa.
- $\text{Gamma} = \text{scale}$ ; It is the range of the decision boundary. The higher its value, the closest are the points that are considered for the decision boundary, the lowest the gamma, the farthest the points are considered for the decision boundary. The value "scale" for gamma means that:

$$\text{gamma} = \left( \frac{1}{\text{number\_of\_features} \times \text{features\_variance}} \right)$$

- $\text{Kernel} = \text{linear}$ ; the kernel is a mathematical function that transforms original data in a higher dimensional space to make the problem linearly separable. It makes SVM able to find a hyperplane that splits the classes also when data is not linearly separable. Linear kernel is the default kernel and uses the data as it is, because they are already linearly separable.

The evaluation metrics of the model are shown in figure 6.2. The confusion matrix is shown in 6.2. Of the three models used, SVM is the one that performs the worst, with an accuracy of 72%, with the same values also for precision, recall and F1-score.

Of the three models used, SVM is the one that performs the worst.

```

Accuracy: 0.7222222222222222
Classification Report:
              precision    recall  f1-score   support

     0       0.73         0.66         0.69         77
     1       0.72         0.78         0.75         85

 accuracy          0.72         0.72         0.72         162
 macro avg         0.72         0.72         0.72         162
 weighted avg      0.72         0.72         0.72         162

```

**Figure 6.2.** SVM metrics**Table 6.2.** SVM confusion matrix

		Predictions	
		0	1
Values	0	51	26
	1	19	66

## 6.3 Gradient boosting

After running GridSearch, we trained the model with the following hyperparameters:

- `N_estimators` = ; represents the number of trees
- `Max_depth` ; it is the maximum depth of the trees
- `Learning_rate`
- `Subsample` ; percentage of samples used for each tree

The evaluation metrics of the model are shown in figure 6.3. The confusion matrix is shown in Tab.6.3. Of the three models, Gradient boosting is the one that performs the best. The model obtains an accuracy score of 98,7% and precision, recall and F1-score values of 99%.

```

Accuracy: 0.9877
Classification Report:
              precision    recall  f1-score   support

     0       0.97         1.00         0.99         77
     1       1.00         0.98         0.99         85

 accuracy          0.99         0.99         0.99         162
 macro avg         0.99         0.99         0.99         162
 weighted avg      0.99         0.99         0.99         162

```

**Figure 6.3.** Gradient boosting metrics

**Table 6.3.** Gradient boosting confusion matrix

		Predictions	
		0	1
Values	0	77	0
	1	2	83



## Chapter 7

# Conclusions

In this work, we investigate malicious practices within the Ethereum blockchain and the feasibility of detecting such events. In particular we focus on "Airdrop scams", a type of scam that is performed after sending multiple transfers to different addresses in the same transaction. We parse the first 1000 event logs of all ERC-20 compliant smart contracts until 2023, collecting different information from which we create several datasets. We found that most of the tokens, around 90%, perform less than 1000 events during their lifetime and that almost all the transactions involving the tokens contain only one transfer, meaning that using airdrop to distribute the token is uncommon.

The next step involved identifying ground truths to label the tokens that performed airdrops, a hard challenge, due to the lack of single source for this information. We relied on various tools to collect the labels, among which we performed manual searches.

We, therefore, introduce a new labeled dataset containing around 1000 tokens with 16 features that we used to train three different Machine Learning algorithms: Random Forest, SVM, and Gradient boosting. We trained each model with the best possible hyperparameters, that we found with the function Gridsearch. The Machine Learning models that we trained yield excellent results in the classification. In particular, we find Gradient boosting to be the best-performing model in terms of metrics, with an accuracy of 98% and a 99% score for precision, recall, and F1-score. Surprisingly, SVM was the worst-performing model with a 72% score for all metrics: accuracy, precision, recall, and F1-score.

Our results show that it is indeed possible to detect potential airdrop-based scams simply relying on the distribution patterns and some associated token attributes.

Throughout this research, I acquired extensive knowledge of Ethereum's mechanisms, transaction processes, and the methods through which certain scams work. Finally, this research highlights the need for caution when interacting with the blockchain, as even transactions and assets that appear harmless may be linked to malicious activities.

# Bibliography

- [1] BREIMAN, L. Random forests. *Machine Learning*, **45** (2001).
- [2] BUTERIN, V. Ethereum: A next-generation smart contract and decentralized application platform. *Ethereum.org*, (2014).
- [3] CERNERA, F., LA MORGIA, M., MEI, A., AND SASSI, F. Token spammers, rug pulls, and sniper bots: An analysis of the ecosystem of tokens in ethereum and in the binance smart chain. *32nd USENIX Security Symposium (USENIX Security 23)*, (2023).
- [4] CHAUDHRY, N. AND YOUSAF, M. M. Consensus Algorithms in Blockchain: Comparative Analysis, Challenges and Opportunities. *University of Punjab*, (2018).
- [5] ETHEREUM. Consensus Mechanisms. Available from: <https://ethereum.org/en/developers/docs/consensus-mechanisms/>.
- [6] ETHEREUM. Proof of Stake (PoS). Available from: <https://ethereum.org/en/developers/docs/consensus-mechanisms/pow/>.
- [7] ETHEREUM. Proof of Work (PoW). Available from: <https://ethereum.org/en/developers/docs/consensus-mechanisms/pow/>.
- [8] ETHEREUM. EVM. Available from: <https://ethereum.org/en/developers/docs/evm/>.
- [9] FORTA. Forta. Available from: <https://forta.org/>.
- [10] FORTA. Forta Datasets. Available from: <https://github.com/forta-network/labelled-datasets/tree/main/labels/1>.
- [11] FRIEDMAN, J. H. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, **29** (2001), 1189. Available from: <http://www.jstor.org/stable/2699986>.
- [12] IBM. Blockchain Use Cases. Available from: <https://www.ibm.com/blockchain/use-cases/>.
- [13] IBM. Smart Contracts. Available from: <https://www.ibm.com/topics/smart-contracts>.

- [14] LA MORGIA, M., MEI, A., MONGARDINI, A. M., AND NEMMI, E. N. A game of nfts: Characterizing nft wash trading in the ethereum blockchain. In *2023 IEEE 43rd International Conference on Distributed Computing Systems (ICDCS)*, pp. 13–24. IEEE (2023).
- [15] LA MORGIA, M., MEI, A., SASSI, F., AND STEFA, J. The doge of wall street: Analysis and detection of pump and dump cryptocurrency manipulations. *ACM Transactions on Internet Technology*, **23** (2023), 1.
- [16] LASHKARI, B. AND MUSILEK, P. A Comprehensive Review of Blockchain Consensus Mechanisms. *IEEE Access*, (2020).
- [17] METAMASK. Metamask-phishing-detect. Available from: <https://github.com/MetaMask/eth-phishing-detect?tab=readme-ov-file>.
- [18] METAMASK. Metamask. Available from: <https://metamask.io/>.
- [19] NAKAMOTO, S. Bitcoin: A peer-to-peer electronic cash system. *Bitcoin.org*, (2008).
- [20] PCMAG. Blockchain: The Invisible Technology That’s Changing the World. Available from: <https://www.pcmag.com/news/blockchain-the-invisible-technology-thats-changing-the-world>.
- [21] PHISHFORT. Phishfort lists. Available from: <https://github.com/phishfort/phishfort-lists>.
- [22] PHISHFORT. Phishfort. Available from: <https://www.phishfort.com/>.
- [23] PROTIĆ, D. AND MIOMIR, S. Anomaly-based intrusion detection: Feature selection and normalization influence to the machine learning models accuracy. *European Journal of Formal Sciences and Engineering*, **3** (2023).
- [24] STITSON, M., WESTON, J. A. E., GAMMERMAN, A., VOVK, V., AND VAPNIK, V. Theory of support vector machines (1996).
- [25] WEBACY. Webacy. Available from: <https://docs.webacy.com/about/overview>.
- [26] YAO, W., YE, J., MURIMI, R., AND WANG, G. A Survey on Consortium Blockchain Consensus Mechanisms. *New Jersey Institute of Technology University of Dallas*, (2021).
- [27] YUAN, Q., HUANG, B., ZHANG, J., WU, J., ZHANG, H., AND ZHANG, X. Detecting phishing scams on ethereum based on transaction records. In *2020 IEEE international symposium on circuits and systems (ISCAS)*, pp. 1–5. IEEE (2020).