





## Article

# ChronoEOS 2.0: Device Fingerprinting and EOSIO Blockchain Technology for On-Running Forensic Analysis in an IoT Environment<sup>†</sup>

José Álvaro Fernández-Carrasco \*, Xabier Echeberria-Barrio , Daniel Paredes-García, Francesco Zola   
and Raul Orduna-Urrutia 

Fundación Vicomtech, Digital Security Department, Basque Research and Technology Alliance (BRTA), Mikeletegi 57, 20009 Donostia-San Sebastián, Spain

\* Correspondence: jafernandez@vicomtech.org

<sup>†</sup> This paper is an extended version of our paper published in 4th International Congress on Blockchain and Applications, L'Aquila, Italy, 13–15 July 2022. ChronoEOS: Configuration control system based on EOSIO blockchain for on-running forensic analysis.

**Abstract:** In industrial environments there are critical devices, so their correct operation must be ensured. In particular, having a secure record of the different events related to these devices is essential. Thus, this record can be used in future forensic investigations in case of accidents or production failures. In this sense, blockchain technology can bring reliability to the event log. In this paper, ChronoEOS 2.0, an extension of ChronoEOS, is presented. This new version can record the events that occur in multiple industrial robotic arms by deploying a Smart Contract in the EOSIO blockchain so that all events are immutably recorded in the blockchain. Furthermore, the new version allows using a unique fingerprint of the robot before registering an event in the blockchain. This fingerprint depends only on the characteristics of the operation and configuration of the robot. For this reason, ChronoEOS 2.0 not only increase the ability of ChronoEOS in terms of handling multiple devices but also increases the security and reliability of the operations. Finally, in this study, we verify that the new improvements have little impact on the hosting resources (RAM and Network are not altered, while CPU consumption is slightly higher due to the device fingerprinting module).

**Keywords:** EOSIO blockchain; forensic analysis; device fingerprint; industrial security



**Citation:** Fernández-Carrasco, J.Á.; Echeberria-Barrio, X.; Paredes-García, D.; Zola, F.; Orduna-Urrutia, R. ChronoEOS 2.0: Device Fingerprinting and EOSIO Blockchain Technology for On-Running Forensic Analysis in an IoT Environment. *Smart Cities* **2023**, *6*, 897–912. <https://doi.org/10.3390/smartcities6020043>

Academic Editors: Miguel Pincheira and Massimo Vecchio

Received: 31 January 2023

Revised: 24 February 2023

Accepted: 27 February 2023

Published: 10 March 2023



**Copyright:** © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

Cybercrime will have associated damage costs of around 10.5 trillion by 2025 (<https://cybersecurityventures.com/hackerpocalypse-cybercrime-report-2016/> accessed on 30 January 2023) (in 2015, it was 3 trillion, three times lower), and this growth is not expected to slow down in the short/medium term. Another fact that reflects the considerable impact of cybercrime is that according to the CyberEdge group report (<https://cyber-edge.com/cdr/> accessed on 30 January 2023), 86% of the organizations analyzed suffered a cyber-attack in the last year (2021).

Although attack methods are becoming more sophisticated and harder to detect [1], many resources, both financial and human, are being invested to find solutions to mitigate the effect of these cyber-attacks or prevent them from occurring in the first place. Particularly in the Internet of Things (IoT) field, the number of attacks is growing since it is a sector in which security standards are still under development [2]. Furthermore, there are usually significant vulnerabilities and the impact can be severe for the environment that suffers the attack [3]. This is due to the fact that it is increasingly common in an industrial environment to find a multitude of interconnected devices, sharing information between them about the production processes being carried out. In this way, an attacker currently does not need to have direct contact with the target device, as it can be connected to the network and proceed with his attack attempts remotely [4].

Because of this, and the large amount of information and data that is handled in this type of environment, it is crucial to have a secure record of the evidence of the use of the different elements of the environment. In this way, this record can be useful in future forensic investigations in the event of an attack, accident or misuse of the equipment [5]. For example, in an environment with several industrial robots in a production plant, any variation in the configuration of the robots can have serious consequences, both in terms of employee safety and in terms of delays and failures in production. Logically, for this evidence repository to be valid in forensic analysis investigations, it is important to ensure the veracity of the information that has been recorded. In other words, it must be ensured that all relevant information has been collected and that it has not been altered at any time by an external agent, and blockchain technology fulfills this requirement [6]. Another problem is that in many cases, in order to extract evidence in an industrial environment, it is necessary to stop production and extract the information offline. This is not desirable, since the ideal is not to affect production.

For this reason, this paper presents ChronoEOS 2.0, a solution based on blockchain technology and device fingerprinting that allows having a record of evidence of the events that occur in an industrial environment [7]. This work is an extension of a conference paper [8], which presented a solution for forensic analysis on a configuration file of a single industrial robotic arm. The main difference with the previous project is that this time the developed solution allows having a decentralized log with the events of several robots. In addition, to ensure the veracity of the information that is recorded, a previous step of univocal identification of robots based on device fingerprinting is performed, so that it is necessary to know the fingerprint of that robot to enter a new event. In this way, an extra layer of security is provided to the system.

### *1.1. Device Fingerprinting Description*

Device fingerprinting [9] involves gathering device information to generate particular signatures that identify individual devices. Those signatures can cover a set of devices or individual devices according to the information they contain and how restrictive it is. Moreover, device fingerprinting can be active if it sends queries to the target device to analyze the response or passive if it only uses a sniffer to capture and analyze traffic but never sends traffic to the target device. An effective device fingerprint must satisfy two main properties, the difficulty of forgery and the stability of the features used and localization [10]. Therefore, classical device identifiers, such as IP, cannot be the unique characteristics of the fingerprint, as they are easily modifiable. In addition, the features that can be modified at the minimum generate fingerprints that are valid for a short time due to their low stability. Particularly, robots are considered devices, and they contain operative systems. Therefore, the device fingerprinting technology can be implemented on them. Moreover, there exist works where the device fingerprinting technology has been modified for the robots, and their particular properties, such as movements or the energy they consume [11].

### *1.2. Blockchain in Forensic Analysis*

According to the National Institute of Standards and Technology (NIST), digital forensics is understood as "The process used to acquire, preserve, analyze, and report on evidence using scientific methods that are demonstrably reliable, accurate, and repeatable such that it may be used in judicial proceedings". Because electronic evidence is present in cybercrimes, digital forensics refers to all evidence and information that can be obtained from electronic devices, whether computers, cell phones, or external storage devices [12], so any item stored digitally can become evidence in a digital forensic investigation [13].

Accessibility to this information by investigators is relatively simple since there is always an electronic trace of the processes performed [14]. However, the treatment of the evidence must be done following a series of specific patterns or steps, since any contamination of the evidence could imply its invalidity in a judicial process. To preserve

this integrity, a process called chain of custody takes place, through which the integrity of the evidence is preserved by keeping an exhaustive and chronological record of all the stages from the time the evidence is obtained until it is presented as criminal evidence. This implies control of the persons who collect the evidence, of the persons or entities that make use of such information, and of a dated record of each of the movements of the evidence. In addition, to avoid contamination of the original evidence, it is necessary to carry out all investigations through a copy of the available information, since this avoids direct processing and, with it, the possible contamination of this information [15].

This chain of custody process can be divided into five main stages, as explained by [16]: Collect Evidence, Authenticate, Examine, Analyze and Report Evidence.

The first stage consists of collecting the samples. It, once the elements that make up this evidence are known, it is necessary to use equipment that safely allows its acquisition, since an error in its extraction could leave the evidence with an invalid character due to its contamination. The second part is related to the authentication of the evidence found. For this, it will be necessary to verify the veracity of the evidence and its origin to be able to accept it. The third stage consists of the thorough examination of all the evidence, i.e., the search for possible hidden information in order not to lose any evidence, as it will be important to be analyzed in the next stage. The fourth stage consists of the analysis of all the information collected. This stage is crucial because the results obtained from the analysis will provide the evidence that proves the guilt or innocence of the alleged cyber attacker. Finally, the fifth stage consists of the preparation of a complete report with the entire investigation process from beginning to end, which will be delivered as final evidence in a trial.

Of all these processes, one of the most complicated phases is the verification of the information, since it is complex to ensure that no information available in the registry has been adulterated or eliminated by cyber-attackers, which would cause the integrity of such information to be violated. It is here where blockchain technology acquires great importance [17], since it is a blockchain-based technology in which transactions are recorded in a public, secure and decentralized manner, where all blockchain participants have access to the database and in which there is no central administrator who is the provider of the information, thus being all the information immutable.

There are several fields of application where this blockchain technology is implemented. Among them is [18], which uses it in forensic applications for connected vehicles, allowing the collection of all available information, such as, for example, own and third-party driving speed, vehicle braking information, the state of the vehicles or the state of the roads, among others, securely and truthfully. Another environment where blockchain technology is being used in IoT environments. An example of this is found in [19], where its implementation based on Hyperledger Fabric is proposed, which is a blockchain authorized to meet privacy requirements being used in smart homes to preserve the integrity and veracity of the data collected. Another implementation framework where this technology is used is the agri-food industry, as presented by [20], which exposes the use of blockchain to perform an exhaustive record of food traceability, thus reliably certifying its origin, from the farm to the arrival of the consumer. Finally, an example of an application in the pharmaceutical industry is [21], which implements this blockchain technology to perform a reliable and secure record of the traceability of the processes carried out in the distribution network of products in the industry.

### 1.3. EOSIO Blockchain

EOSIO is a type of blockchain that emerged in 2018 from the hand of the company block.one (ref white paper) and is specially designed to facilitate the operation and deployment of Decentralized Applications (dApps), for the creation of Smart Contracts [22].

### 1.3.1. Consensus Protocol

Unlike other types of blockchain, such as Bitcoin [23] or Ethereum 1.0 [24], which use the Proof of Work (PoW) consensus protocol, in EOSIO the protocol is the Delegated Proof of Stake (DPoS). This protocol provides better performance, both in terms of energy consumption and the number of transactions per second that can be recorded. Thus, in EOSIO, up to 8000 transactions per second (TPS) can be performed, unlike the 7 TPS that can be performed in Bitcoin or 11 TPS in Ethereum [24].

In EOSIO [25], 21 producer nodes are elected through a voting system in which stakeholders can participate. These nodes share the block's production, so that each producer signs 12 blocks, with a period of 0.5 seconds between blocks. This approach enables the possibility to work in environments with a very quick response, and, at the same time, keep control over the block production. Therefore, each producer node generates blocks for 6 s in each round. These parameters (12 blocks and 0.5 s production time of each block) are constants of EOSIO layer 1 and cannot be changed. However, the number of producers can be different than 21. PoW consumes a large number of resources, which translates into problems related to the pollution this causes, provoking criticism from many sectors. On the other hand, in PoS (Proof of Stake) [26] and DPoS, the consensus is reached by dedicating tokens from the blockchain (staking them). So, the more tokens staked [27], the more importance they gain in the participation of consensus (in PoS) and the more weight they gain in voting for the producers (in DPoS). In EOSIO, BFT and DPoS protocols are employed at different network layers (<https://eos.io/news/dpos-bft-pipelined-byzantine-fault-tolerance/> accessed on 30 January 2023):

- **Layer 1: Asynchronous Byzantine Fault Tolerance (aBFT).** Its main function is to monitor and confirm the execution of the blocks so that they are permanently recorded in the blockchain. Thus, aBFT receives a list (schedule) of block-producing nodes, which have been proposed in the second layer (DPoS), and checks that the blocks have been produced correctly through a confirmation process. This process requires two-thirds of the producers to confirm or validate each block twice before considering it irreversible.
- **Layer 2: Delegated Proof of Stake (DPoS).** In this layer, block-producing nodes are elected to be authorized to sign blocks in the network [28]. Thus, in the EOSIO main network, 21 producer nodes are chosen, which will be in charge of producing the blocks.

### 1.3.2. EOSIO's Architecture

EOSIO works with three key concepts, which define its architecture [22]:

- **RAM:** Storage space (in bytes) where the blockchain stores information. Using RAM for storage allows for faster queries.
- **CPU:** Processing capacity of an account (in microseconds) and represents how much processing time an account has when a transaction is executed.
- **NET:** Measures (in bytes) the size of the transaction stored in the blockchain.

When a transaction is executed, NET and CPU are consumed by the account executing the transaction, so that account must have the necessary resources.

CPU and NET are mostly used by ordinary users to interact with the blockchain's applications and contracts, while RAM is used almost exclusively by developers for implementing decentralized applications (dApps) [22].

### 1.3.3. Multi Index Tables in EOSIO

Multi Index Tables in EOSIO [29] allow to execute, create, read, update and delete operations (CRUD operations), while typical blockchain operations are only created and read. This is why they are a quite useful type of structure in some Smart Contracts, as it provides a quick way to store information related to the application. In this way, although the transactions generated by the use of the Smart Contract are stored in the blockchain, in

the Multi Index Table there is a direct record of the data to be stored. This type of table is stored in EOSIO's RAM cache.

Tables in EOSIO are stored according to the following three parameters:

- **code:** name of the account that first displayed the Smart Contract where the table structure is defined.
- **scope:** basically the name (title) of the table.
- **table:** name of the table type.

For example, if you want to query a table in the blockchain with code="user10", scope="r2", table="robotEvent", it means that the account that has deployed that contract (the account that owns the table) is "user10", the table is of type "robotEvent" and the table being queried is called "r2".

## 2. Materials and Methods

### 2.1. Use Case Description

Fernández et al. [8] present a novel method for performing the forensic analysis on-running. At the same time, it may be used to detect anomalies in configuration modifications, preventing incidents and failures that need to be investigated. They mention that industrial forensic analysis is dominated by techniques based on evaluating methods once the malicious modifications are carried out. Moreover, they argue that those after-methods do not avoid stopping industrial systems from generating losses in production-chain. Those reasons made them develop a novel forensic analysis method called ChronoEOS.

This work introduces ChronoEOS 2.0, a new approach for complexing and securing the method presented to ChronoEOS without affecting the hosting resources. This new version of ChronoEOS is more complex and scalable due to the capability to incorporate several robots simultaneously in the deployed traceability system. On the other hand, security is obtained by incorporating a new module to generate a device fingerprinting signature for each participating robot, which is essential to modify the traceability system.

In this way, to make any modification in the event log, it is necessary to provide two keys, thus having a double security factor of the application:

- The private key [30] of the account that deployed the Smart Contract. In this use case, there is an administrator account, which is in charge of making the calls to the EOSIO blockchain to incorporate new events in the tables.
- The fingerprint of the device in question for which a new event is to be registered in the blockchain.

In this case, ChronoEOS 2.0 is implemented in a concrete environment formed by robotic arms, detailed in Section 3.1, for monitoring the new events that they generate due to the new movements or different actions.

### 2.2. Universal Robots

Universal Robots (UR) Ltd. is an American company of Norwegian origin dedicated to the manufacturing of collaborative robotic arms (Cobots) for industrial environments, whether automotive, metallurgy, electronics, or food, allowing the automation and optimization of repetitive industrial processes. UR has a simulation software of these robots used for offline programming, called URSim, created to be used in the Linux operating system. However, URSim allows the possibility of being executed on a Windows server, provided that a virtual machine is used for this purpose.

This project is going to be developed using URSim simulation software, more specifically in URSim version 3.14.3. This simulator has some limitations concerning real robotic arms. While the simulation of digital I/O is possible, some functionalities differ from the real ones, such as the fact that the force mode does not work, there is no emergency stop, it always performs perfect trajectories, or the fact that collisions with itself cannot occur.

Among the robotic arms to be used are the UR3, UR5 and UR10. All the robots have six axes, with a rotation of 360°, except for the last axis of the UR3e, which has an infinite



rotation. The main differences between the various versions lie in the dimensions, weight, payload and reach of the different robots. The UR3 series of robots is the smallest, most compact and lightest, making it ideal for use on work tables in small spaces. The UR5 series has medium dimensions, with a slightly higher weight than the previous one and a longer reach, allowing greater versatility, which allows it to adapt to a more significant number of work environments. Finally, the UR10 series of robots is the largest while allowing greater reach and payload, making them ideal for work environments related to machinery maintenance. In addition, all robots are equipped with a programming console that allows the control of the robots and their programming (<https://www.universal-robots.com/> accessed on 30 January 2023). A summary of the main features of the robotic arms is shown in Table 1.

**Table 1.** UR Robots characteristics.

		UR3	UR5	UR10
<b>Footprint</b>		128 mm	149 mm	190 mm
<b>Payload</b>		3 kg	5 kg	10 kg
<b>Reach</b>		500 mm	850 mm	1300 mm
<b>Weight</b>		11.0 kg	18.4 kg	28.9 kg
<b>Speed Movement</b>	<b>Base</b>	$\pm 180^\circ/\text{s}$	$\pm 180^\circ/\text{s}$	$\pm 120^\circ/\text{s}$
	<b>Shoulder</b>	$\pm 180^\circ/\text{s}$	$\pm 180^\circ/\text{s}$	$\pm 120^\circ/\text{s}$
	<b>Elbow</b>	$\pm 180^\circ/\text{s}$	$\pm 180^\circ/\text{s}$	$\pm 180^\circ/\text{s}$
	<b>Wrist 1</b>	$\pm 360^\circ/\text{s}$	$\pm 180^\circ/\text{s}$	$\pm 180^\circ/\text{s}$
	<b>Wrist 2</b>	$\pm 360^\circ/\text{s}$	$\pm 180^\circ/\text{s}$	$\pm 180^\circ/\text{s}$
	<b>Wrist 3</b>	$\pm 360^\circ/\text{s}$	$\pm 180^\circ/\text{s}$	$\pm 180^\circ/\text{s}$

The working environment in which this project will be developed consists of an industrial plant with 6 Cobots robotic arms, two of them from the UR3 series, two others from the UR5 series, and the last two from the UR10 series.

### 2.3. Device Fingerprinting

The improvements of the solution in an environment containing multiple robots generated the need for an identifier technology to distinguish and represent those robots in the blockchain. The device fingerprinting technology for robots was the decided solution because of the advantages it gives from the security side. This way, only the authentic robot or the administrator that knows all the features of the original robots can introduce new information in the event tracking system.

Device fingerprinting generates the signatures that represent the robots taking into account the features that make the robots unique robots. In particular, in this work, the presented use case works with specific robots introduced in Section 2.2. Moreover, Section 2.3 compares the differences between the robot types participating in this use case. Via this comparison and checking the configuration files of the robots, several parameters were taken to generate the mentioned identification signature. Table 2 describes those parameters; even it details the name of the file where they can be found.

Section 2.2 mentioned that six joints form each robot, and the parameter in Table 2 are features of them. Therefore, the robots have a vector of features per each parameter, concretely, six values. Notice that those parameters would be the same in the case of two same-model robots. In that case, with those selected parameters, both robots would share the same device fingerprinting signature; therefore, the developed system does not identify the robots correctly. Because of that, another parameter is necessary to differentiate the robots that share the model. In this case, the decided extra parameter was the IP of the robot, and then the signature contained both the IP and the technical information about the

robot. Table 3 shows the values of the selected parameters for each type of robot, i.e., the values that the same robot models share.

**Table 2.** This table shows the parameters that are considered to generate the device fingerprint of the robots.

Parameter	Description
Mass [kg]	The weight of each joint that forms the robot.
Center of Mass [m]	The unique point where the weighted joint's relative position of the distributed mass sums to zero.
Rotor Inertia [ $\text{kg} \times \text{m}^2$ ]	A scalar value which tells us how difficult a joint is to change the rotational velocity of the object around a given rotational axis.
Winding Resistance [ $\Omega$ ]	The resistance of a length of joint's copper wires from one end to the other.
Winding Inductance [H]	The tendency of an electrical conductor to oppose a change in the electric current flowing through its joint.
Maximum Velocity [ $\frac{\text{m}}{\text{s}}$ ]	The maximum speed that a robot's joint can reach.

**Table 3.** This table shows the values that the parameters introduced in Table 2 obtain according to the robot model.

Parameter	UR3	UR5	UR10
Mass	[2, 3.42, 1.26, 0.8, 0.8, 0.35]	[3.7, 8.393, 2.33, 1.219, 1.219, 0.1879]	[7.1, 12.7, 4.27, 2, 2, 0.365]
Center of Mass	[0, −0.02, 0], [0.13, 0, 0.1157], [0.05, 0, 0.0238], [0, 0, 0.01], [0, 0, 0.01], [0, 0, −0.02]]	[0, −0.02561, 0.00193], [0.2125, 0, 0.11336], [0.15, 0, 0.0265], [0, −0.0018, 0.01634], [0, 0.0018, 0.01634], [0, 0, −0.001159]	[0.021, 0, 0.027], [0.38, 0, 0.158], [0.24, 0, 0.068], [0, 0.007, 0.018], [0, 0.007, 0.018], [0, 0, −0.026]
Rotor Inertia	$[6 \times 10^{-5}, 6 \times 10^{-5},$ $2.0767 \times 10^{-5}, 7 \times 10^{-6},$ $7 \times 10^{-6}, 7 \times 10^{-6}]$	$[187.74 \times 10^{-6}, 187.74 \times 10^{-6},$ $187.74 \times 10^{-6}, 2.0767 \times 10^{-5},$ $2.0767 \times 10^{-5}, 2.0767 \times 10^{-5}]$	$[7 \times 10^{-4}, 7 \times 10^{-4}, 187.74 \times 10^{-6},$ $6 \times 10^{-5}, 6 \times 10^{-5}, 6 \times 10^{-5}]$
Winding Resistance	[0.78, 0.78, 1.65, 2.394, 2.394, 2.394]	[0.3, 0.3, 0.3, 1.65, 1.65, 1.65]	[0.088, 0.088, 0.3, 0.78, 0.78, 0.78]
Winding Inductance	$[15 \times 10^{-4}, 15 \times 10^{-4},$ $25 \times 10^{-4}, 13 \times 10^{-4},$ $13 \times 10^{-4}, 13 \times 10^{-4}]$	$[83 \times 10^{-5}, 83 \times 10^{-5}, 83 \times 10^{-5},$ $25 \times 10^{-4}, 25 \times 10^{-4}, 25 \times 10^{-4}]$	$[33 \times 10^{-5}, 33 \times 10^{-5}, 83 \times 10^{-5},$ $15 \times 10^{-4}, 15 \times 10^{-4}, 15 \times 10^{-4}]$
Maximum Velocity	[3.3416, 3.3416, 3.3416, 6.4832, 6.4832, 6.4832]	[3.3416, 3.3416, 3.3416, 3.3416, 3.3416, 3.3416]	[2.2944, 2.2944, 3.3416, 3.3416, 3.3416, 3.3416]

Those values and the respective IP are concatenated and flattened, obtaining a long vector with all the corresponding values. This vector is injected in a sha-256 algorithm that computes the signature identifying the respective robot. The generated signature is private, and it is necessary to update the traceability of the robot's events, i.e., only the robot, which contains the essential information to generate the signature, and the administrator, who knows the signature of the robot. The smart contract [31] to create this scenario is detailed in Section 3, which allows maintaining the generated signature private and running it if the petition contains the adequate signature.

### 3. Results

This section details the presented system ChronoEOS 2.0, concretely describes each module of the system, and how those modules combine and their disposition. Moreover, this system is evaluated in a concrete environment according to the consumed resources in RAM, CPU, and DISK.

### 3.1. ChronoEOS 2.0

In this paper, ChronoEOS 2.0 is developed, deployed, and evaluated. It tracks the events generated according to the actions made by an industrial robot. In this way, it allows analyzing the activities of the targeted robot in streaming; even the blockchain provides the trust of the event registered there due to the invulnerability it gives to the information it contains. The main difference between ChronoEOS 2.0 and the system presented by Fernández et al. in [8], apart from the functionality of the system and the complexity is a new device fingerprinting module introduced to increase the security of the system. ChronoEOS 2.0 consists of the following modules: Device Fingerprinting, Lookout Agent, Rest API, and Blockchain.

*Device Fingerprinting module* computes the fingerprint introduced in Section 2.3, which consists of a hash signature. It is calculated using the configuration files looked at by the module. Moreover, this module introduces the target robot's IP in the computation to distinguish the signatures of the same types of robots since they have the same configurations. The algorithm used to generate those signatures is sha-256, which gives a hexadecimal hash according to the received input. That is, the generated fingerprints are unique per robot, and because of that, they identify the respective robots. Therefore, the device fingerprinting module looks at the desired configuration files and IP, computes the identification signature through mentioned information, and outputs that fingerprint to the lookout agent module.

*Lookout Agent module* is deployed in the industrial robot and is in charge of detecting the new events generated according to the activities of the targeted robot. This code is executed in a scheduled process (cron job), so the events are saved with the timestamp when the target robot runs the action. Identifying the robot is essential to register new activity, and it is the signature generated by the Device fingerprinting module. That module passes Lookout Agent the computed signature, in addition to the private key of the administrator account, which has deployed the Smart Contract, allowing it to record the newly detected events.

*Rest API module* is the interface where the operator (authorized user) can interact with both the Smart Contract and the target information. The methods available in the API can be divided into two sub-blocks: Blockchain EOS and Smart Contract. The first one contains the two necessary methods to start and stop the production of blocks in EOSIO, another for displaying the tracking of the targeted robot's activities, showing the registered events, and a method to obtain information on the transactions that have been recorded. The second one contains functions to display the information for a given account and a method to call the upsert function of the Smart Contract. This function registers desired events in the blockchain.

*Blockchain module*, as mentioned previously, is used as an event repository, giving the immutable property essential for the trustworthiness of traceability. Concretely, ChronoEOS 2.0 implements EOSIO private blockchain with three block-producing nodes. Moreover, an API node and a seed node are incorporated. The API node listens to HTTP requests to the blockchain, while the seed node stores in memory the information of all the blocks in the case of any blockchain crash to recover it. Due to its immutability, the blockchain is the key element in this application to ensure the recording of evidence and to carry out forensic analysis with guarantees in this industrial environment. Otherwise, the desired functionalities are incorporated into the mentioned blockchain by deploying a Smart Contract. It uses multi-index tables to register the target information, which is a way of caching status or data in RAM for fast access. The blockchain records the transactions, but the multi-index tables allow the application data to be stored. Each robot has a particular table with the event logs. In particular, in the Smart Contract deployed in ChronoEOS 2.0, for each registered activity, the following information is stored:

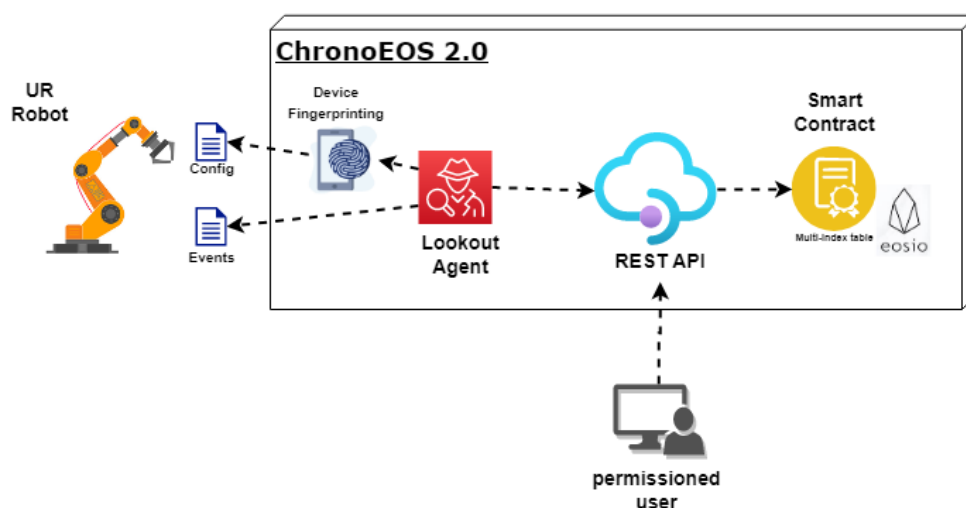
1. *Event Number*: The number of the registered event.
2. *Timestamp*: Timestamp of the registered event.
3. *Type*: The type of the registered event. For example, it may be INFO, WARNING, and so on.



4. *Event*: The new event generated by the targeted robot.

That Smart Contract incorporates an upsert function, which receives the mentioned information as inputs, allowing the event's registration in the table. The event number is obtained directly in the function, while the timestamp and type are given with the event information. An important feature added to the contract is that the upsert function can only be called by the entity that knows the fingerprint of the respective robot or the robot itself and the private key of the account that first deployed the Smart Contract. For this reason, the other entities, which do not know the fingerprint of the robot and the private key, would not be able to add new events in the blockchain, making the proposed system more secure.

Figure 1 shows how ChronoEOS 2.0 incorporates the presented modules. The lookout agent module constantly checks if any new event registration is generated. In case of any activity is detected, the lookout agent module asks for the targeted robot's fingerprint to the device fingerprinting module. It consults the configuration files of the target robot and takes the values of the concrete features of the target robot defined in Section 2.3. The device fingerprinting module computes the signature, using obtained values and the target robot's IP, and sends it to the lookout agent module. Once the lookout agent module receives the new events, the private key and the robot fingerprint, it calls the upsert function of the Rest API module. This function of the rest API module activates the function upsert incorporated previously in the blockchain module via the deployed Smart Contract.



**Figure 1.** The diagram of ChronoEOS 2.0, considering the simplest scenario.

### 3.2. Evaluation

The environment created to deploy the ChronoEOS 2.0 is formed by six industrial robots mentioned in Section 2.2, two by type. Concretely, Table 4 details the IP and the robot type of each robot participating in the environment.

**Table 4.** This table shows the type and IP of each robot according to the robot ID.

Robot ID	Type	IP
R0	UR3	192.168.20.10
R1	UR5	192.168.20.15
R2	UR10	192.168.20.12
R3	UR3	192.168.20.3
R4	UR5	192.168.20.18
R5	UR10	192.168.20.7

Once those properties were fixed and looking at the configurations presented in Section 2.3 by robot type, the fingerprint of each robot can be computed. Therefore, the following signatures are the fingerprints of each robot participating in the environment:

- R0: 0ba67836a6ca85dbf5a09f09de667fb790bc4f3a54eb4b3b2b1d1030d31f76b4
- R1: f282b657121fa12c7b89c0ee9c5f65a42b394b544ba735a817031e6ed228c34b
- R2: 22a05b03c0abd15b4862f89362f1a77cad55dc90862c3ef856879cc06274ec4b
- R3: c2da0202b891511b9d64c373d571d2056169137f9197f1f5b7481658ebfced45
- R4: 0d73df4b99813ebbf475386edab6b28c4abfafbdae25dde99d802e204934f36
- R5: 5bccbcbde9fb780aa90785e7e1012f29e0b2eefa7dc783cbc9893d0e9cc9546b

Those signatures were incorporated into the Smart Contract, making them essential to update the traceability of the respective robot. Therefore, once those fingerprints were introduced in the Smart Contract, it was deployed in the Blockchain module, giving the desired functionalities to it. The details of the Smart Contract are mentioned in Section 3.1. After that, the six lookout agent modules, one per robot, were deployed, and each was responsible for recording the newly generated events of each robot. The presented environment is shown in Figure 2.

The steps required to deploy ChronoEOS 2.0 are presented below:

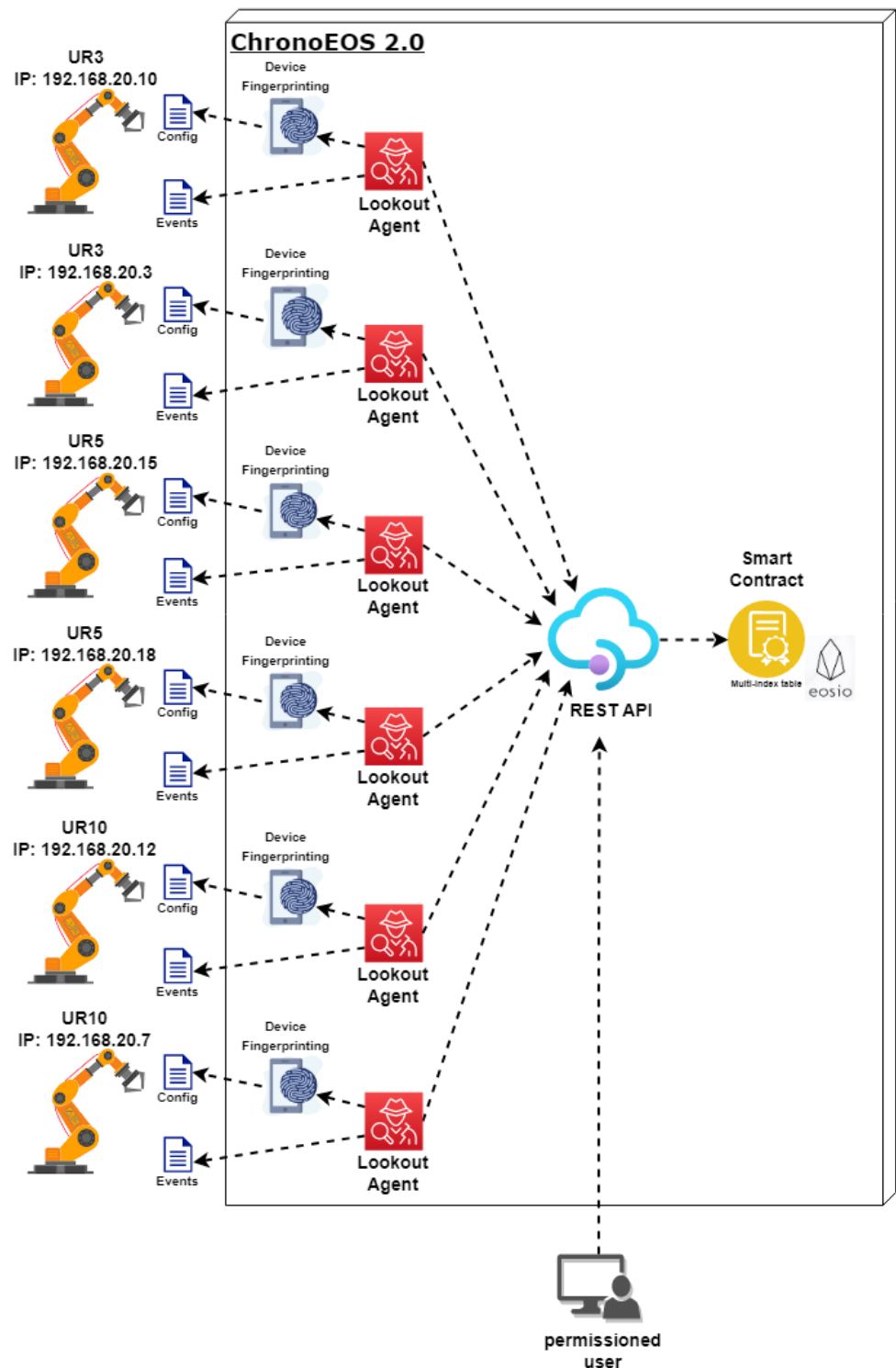
1. To deploy a private blockchain based on EOSIO, with three producer nodes, one HTTP node and one Seed node.
2. To create the account that will deploy the Smart Contract. This account will possess a private key, which should not be shared with others, as it is needed to make changes to the event tables.
3. To deploy the Smart Contract using the account created in the previous step, adding the fingerprints of the six robots in the environment.
4. To deploy the REST API on a secure server, accessible to the robots on which the events are to be stored.
5. To deploy the Lookout Agent for each robot so it can access the information needed to generate the robot's fingerprint and the event log. This will also allow calling the upsert function of the Smart Contract.
6. To deploy in each robot environment the Device Fingerprinting Module, so that the Lookout Agent can access the fingerprint of each robot when a new event must be added to the blockchain.
7. Execute the Lookout Agent script in a scheduled process (cron, for example).

To evaluate the performance of ChronoEOS 2.0, a strategy similar to that used in the previous version [8] was followed. In this way, the resources consumed by the tool in terms of CPU, RAM and network packet transmission when calling the API have been measured. It is important to remember that it is critical that the application does not require high resource consumption since the equipment and devices in an industrial environment usually have limited and exclusive resources for the execution of their tasks. That is why it is necessary that the application is as light as possible, otherwise it could not be deployed. In addition, the memory required by the blockchain to store the generated information has been measured. The Lookout Agent has been installed as a cron process that runs every minute. In addition, the *psutil* python library has been used to obtain the resources consumed by the running processes. Thus, the calls to the Lookout Agent are in seconds (see Figures 3–6): 20, 80, 140, 200, 260.

The graphs show how in each minute there are different peaks in the resources consumed, especially in CPU and packets received. This is due to the fact that when the Lookout Agent is executed, it must query the blockchain for the table data, observe in the robot's event file if new events have occurred, generate a fingerprint for the robot and call the upsert function for each new event.

Compared to the first version of ChronoEOS, the CPU consumption is significantly higher when running, but not excessive. The reason is mainly that in this new version fingerprint generation is added, which requires several operations and calculations. In

addition, it can also be observed how the received data peaks suffer more variations than in the previous version. This is due to the fact that in the one-minute interval between one Lookout Agent execution and another, the number of new events generated by the robot is more disparate. For example, in the last iteration, the peak of data received is more than double the previous peak, because between one peak and the other, the new events that have occurred are very different (more in the last peak).



**Figure 2.** The diagram of the selected environment, where ChronoEOS 2.0 was deployed to be evaluated.

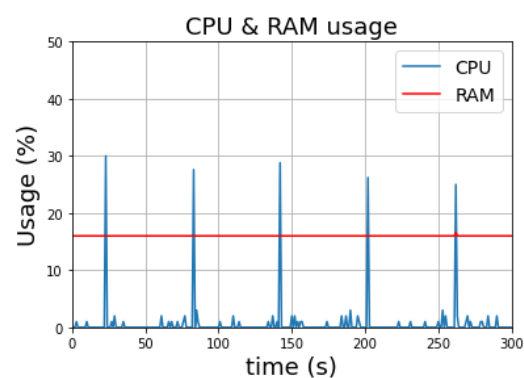


Figure 3. CPU & RAM usage for ChronoEOS 2.0.

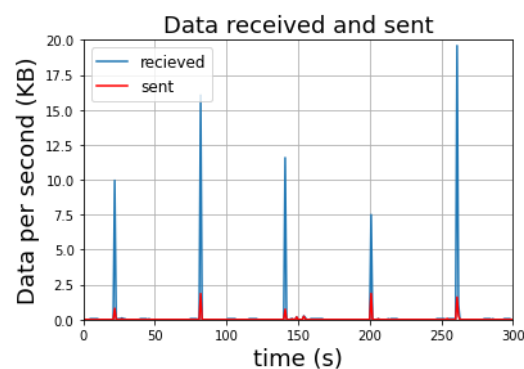


Figure 4. Data I/O for ChronoEOS 2.0.

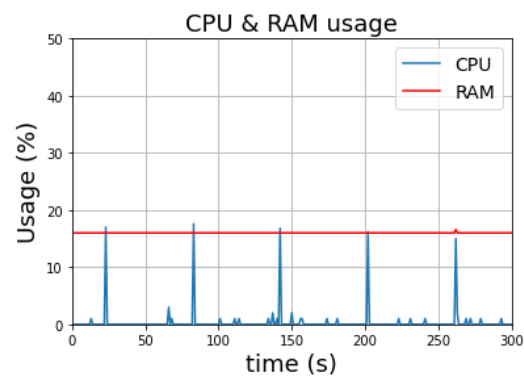


Figure 5. CPU & RAM usage for ChronoEOS.

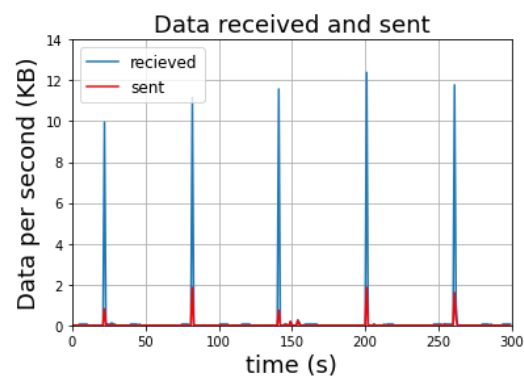


Figure 6. Data I/O for ChronoEOS.

#### 4. Discussion

The results and tests carried out, together with those performed on the first version of ChronoEOS, demonstrate the validity of this system to complete the traceability of events occurring in an industrial environment. Thus, ChronoEOS 2.0 allows a secure, immutable and reliable record of the events that occur in multiple robotic arms in an industrial environment.

These security and immutability features are mainly provided by using blockchain technology, EOSIO in this case. However, the modifications and the design made allow providing an extra layer of security, which makes ChronoEOS a reliable tool in Forensic Analysis procedures. This is so for two main reasons:

1. Because of the structure of the Smart Contract that records the events, which only allows new modifications to the account that has deployed the Smart Contract since it is necessary to have his private key to call the upsert function.
2. For the inclusion of the Device Fingerprinting module. In case an attacker wants to make malicious use of the system and has the private key of the admin account, he will not be able to make changes in the blockchain without the Fingerprint of the robot he wants to modify. In other words, there is a double security factor, unlike the first version of ChronoEOS, which only checks the private key.

Compared to the previous version of ChronoEOS, the improvement in terms of application security is clear, as the Device Fingerprinting module adds an extra layer of security that is difficult for a potential attacker to overcome.

On the other hand, regarding the results obtained, in relation to the consumption of system resources, it is observed that the results are similar to those obtained with the previous version. Thus, it is shown that the application is lightweight, which is a very important aspect in industrial environments. This is so because if the application is to be deployed in the same robot software, it is necessary that it must interfere minimally with the performance of the robot, i.e., not occupy system resources (which are usually limited) that the robot may need to perform its tasks.

As an aspect to be improved to make the tool more reliable and with better performance, it is desirable to improve the security of the blockchain itself. In other words, the EOSIO blockchain must be protected so that it does not suffer unexpected crashes or attacks that could compromise the persistence of the data recorded there. For this reason, it is proposed for future versions choose another blockchain technology with greater intrinsic security, since this is the most important aspect when choosing one type of blockchain or another. Aspects such as the speed of transactions are not particularly critical in forensic analysis tasks.

Another important problem to be solved in the near future is the management of the private key of the users with permission to use the application, i.e., those for whom the Smart Contract has been deployed on the blockchain. The private key is an essential parameter in the security of the system since it is one of the two methods to ensure that ChronoEOS is not being used fraudulently, together with the Device Fingerprinting module.

Currently, from the LookOut Agent, calls to the EOSIO blockchain API are made by inserting the private key in the body of the call. This can be a weak point of the application since if the call is intercepted, the attacker could know the private key of this user. For this reason, we are considering adding a more secure method to make API calls in the future, instead of using the private key. The option proposed is to deploy an IdP (Identity Provider), which allows managing permissions to make API calls, applying some authorization protocol, such as OAuth2, which can be very useful in this use case. Thus, as a header in the API call, an authorization token would have to be introduced, provided by the IdP and applying the OAuth2 protocol.



## 5. Conclusions

The main conclusion that can be drawn from this project is that the inclusion of the Device Fingerprinting module provides an extra layer of security compared to the first version of ChronoEOS. By applying Device Fingerprinting, any attacker who wants to modify the EOSIO blockchain to add or change the events of the robot must know both the private key of the administrator's account and the robot's fingerprint. In addition, this new version has been tested in a more complex industrial environment, with six industrial robots, unlike the first application, which was only applied in an environment with a single robot. Another point where ChronoEOS is improved is that, although the Lookout Agent is still running in a cron process, there is no chance of missing new events, as the timestamp of the last event recorded in the blockchain is always compared with the timestamp of the new events generated by the robot.

With this, ChronoEOS 2.0 is an application that can be used in an industrial environment for Forensic Analysis with guaranteed security and reliability. In addition, the type of blockchain used (EOSIO) is fully adapted to this type of task, thanks mainly to its high block production rate, which allows events to be recorded quickly. The high block production rate is mainly due to the fact that it uses Delegated Proof of Stake (dPOS) as a consensus protocol.

On the other hand, as in the first version of ChronoEOS, the REST API provides significant ease of use and flexibility for authorized users. From the REST API, it is possible to interact with the blockchain quickly to deploy Smart Contracts, resume/stop block production, query the status of the blockchain, and consult tables.

Moreover, despite introducing a new module to calculate the robot's fingerprint every time a new event is introduced in the blockchain, the resources consumed (CPU, RAM and network packet transmission) are not altered compared to the results obtained with the first version of ChronoEOS. Only the CPU consumption is altered, although it is far from being worrying for the correct performance of the system. A slight increase in CPU consumption is sacrificed in exchange for the increased security offered by the addition of the fingerprinting module.

As for future lines of improvement, the aim is to continue fine-tuning the different elements of the application. In particular, the blockchain technology on which to work could be changed. In other words, instead of creating the ecosystem on EOSIO, we could work on another blockchain, such as Ethereum, and analyze the differences in performance between the two. Another possible point of improvement would be to try to develop a module that detects attempted attacks on the system. That is, to have a system that analyses attempts to modify blockchain events and tries to predict the attack in order to take solutions before it happens.

**Author Contributions:** Conceptualization, J.Á.F.-C., X.E.-B. and F.Z.; Introduction, J.Á.F.-C. and X.E.-B.; Materials and Methods, J.Á.F.-C., X.E.-B. and D.P.-G.; Results, J.Á.F.-C.; Discussion, J.Á.F.-C., X.E.-B. and D.P.-G.; Writing—original draft preparation, J.Á.F.-C.; Writing—review and editing, J.Á.F.-C., X.E.-B. and D.P.-G.; Supervision, R.O.-U. and F.Z. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding

**Data Availability Statement:** No new data were created or analyzed in this study. Data sharing is not applicable to this article.

**Acknowledgments:** This work has been partially supported by the Basque Country Government under the ELKARTEK program, project REMEDY (KK-2021/00091), and by the Spanish Centre for the Development of Industrial Technology (CDTI) under the project ÉGIDA (EXP 00122721/CER-20191012).

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Gurunath, R.; Agarwal, M.; Nandi, A.; Samanta, D. An Overview: Security Issue in IoT Network. In Proceedings of the 2018 2nd International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Palladam, India, 30–31 August 2018; pp. 104–107. [\[CrossRef\]](#)
2. Mishra, N.; Pandya, S. Internet of things applications, security challenges, attacks, intrusion detection, and future visions: A systematic review. *IEEE Access* **2021**, *9*, 59353–59377. [\[CrossRef\]](#)
3. Jović, M.; Tijan, E.; Aksentijević, S.; Čišić, D. An Overview of Security Challenges of Seaport IoT Systems. In Proceedings of the 2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), Opatija, Croatia, 20–24 May 2019; pp. 1349–1354. [\[CrossRef\]](#)
4. Kuang, B.; Fu, A.; Susilo, W.; Yu, S.; Gao, Y. A survey of remote attestation in Internet of Things: Attacks, countermeasures, and prospects. *Comput. Secur.* **2022**, *112*, 102498. [\[CrossRef\]](#)
5. Servida, F.; Casey, E. IoT forensic challenges and opportunities for digital traces. *Digit. Investig.* **2019**, *28*, S22–S29. [\[CrossRef\]](#)
6. Zheng, Z.; Xie, S.; Dai, H.; Chen, X.; Wang, H. An Overview of Blockchain Technology: Architecture, Consensus, and Future Trends. In Proceedings of the 2017 IEEE International Congress on Big Data (BigData Congress), Boston, MA, USA, 11–14 December 2017; pp. 557–564. [\[CrossRef\]](#)
7. Al-Khateeb, H.; Epiphaniou, G.; Daly, H. Blockchain for modern digital forensics: The chain-of-custody as a distributed ledger. In *Blockchain and Clinical Trial: Securing Patient Data*; Springer: Cham, Switzerland, 2019; pp. 149–168.
8. Fernandez-Carrasco, J.A.; Egues-Arregui, T.; Zola, F.; Orduna-Urrutia, R. ChronoEOS: Configuration Control System Based on EOSIO Blockchain for On-Running Forensic Analysis. In *Blockchain and Applications, Proceedings of the 4th International Congress, L'Aquila, Italy, 13–15 July 2022*; Springer: Berlin/Heidelberg, Germany, 2023; pp. 37–47.
9. Xu, Q.; Zheng, R.; Saad, W.; Han, Z. Device Fingerprinting in Wireless Networks: Challenges and Opportunities. *IEEE Commun. Surv. Tutorials* **2016**, *18*, 94–104. [\[CrossRef\]](#)
10. Sharaf-Dabbagh, Y.; Saad, W. On the authentication of devices in the Internet of things. In Proceedings of the 2016 IEEE 17th International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM), Coimbra, Portugal, 21–24 June 2016; pp. 1–3. [\[CrossRef\]](#)
11. Pu, H.; He, L.; Zhao, C.; Yau, D.K.; Cheng, P.; Chen, J. Detecting replay attacks against industrial robots via power fingerprinting. In Proceedings of the 18th Conference on Embedded Networked Sensor Systems, Virtual Event, Japan, 16–19 November 2020; pp. 285–297.
12. Divith Devaiah, M.; Metre, P.B. Survey on current Digital forensic practicess. *Int. J. Comput. Eng. Res. Trends* **2017**, *4*, 180–184.
13. Beebe, N. Digital forensic research: The good, the bad and the unaddressed. In Proceedings of the Advances in Digital Forensics V: Fifth IFIP WG 11.9 International Conference on Digital Forensics, Orlando, FL, USA, 26–28 January 2009; Revised Selected Papers 5; Springer: Berlin/Heidelberg, Germany, 2009; pp. 17–36.
14. Stoyanova, M.; Nikoloudakis, Y.; Panagiotakis, S.; Pallis, E.; Markakis, E.K. A survey on the internet of things (IoT) forensics: Challenges, approaches, and open issues. *IEEE Commun. Surv. Tutorials* **2020**, *22*, 1191–1221. [\[CrossRef\]](#)
15. Pollitt, M. A history of digital forensics. In Proceedings of the Advances in Digital Forensics VI: Sixth IFIP WG 11.9 International Conference on Digital Forensics, Hong Kong, China, 4–6 January 2010; Revised Selected Papers 6; Springer: Berlin/Heidelberg, Germany, 2010; pp. 3–15.
16. Giova, G. Improving chain of custody in forensic investigation of electronic digital systems. *Int. J. Comput. Sci. Netw. Secur.* **2011**, *11*, 1–9.
17. Hofmann, F.; Wurster, S.; Ron, E.; Böhmecke-Schwafert, M. The immutability concept of blockchains and benefits of early standardization. In Proceedings of the 2017 ITU Kaleidoscope: Challenges for a Data-Driven Society (ITUK), Nanjing, China, 27–29 November 2017; IEEE: New York, NY, USA, 2017; pp. 1–8.
18. Cebe, M.; Erdin, E.; Akkaya, K.; Aksu, H.; Uluagac, S. Block4forensic: An integrated lightweight blockchain framework for forensics applications of connected vehicles. *IEEE Commun. Mag.* **2018**, *56*, 50–57. [\[CrossRef\]](#)
19. Brotsis, S.; Kolokotronis, N.; Limniotis, K.; Shiaeles, S.; Kavallieros, D.; Bellini, E.; Pavuë, C. Blockchain solutions for forensic evidence preservation in IoT environments. In Proceedings of the 2019 IEEE Conference on Network Softwarization (NetSoft), Paris, France, 24–28 June 2019; IEEE: New York, NY, USA, 2019; pp. 110–114.
20. Wang, Z.; Wang, L.; Xiao, F.; Chen, Q.; Lu, L.; Hong, J. A traditional chinese medicine traceability system based on lightweight blockchain. *J. Med. Internet Res.* **2021**, *23*, e25946. [\[CrossRef\]](#) [\[PubMed\]](#)
21. Chiacchio, F.; Compagno, L.; D'Urso, D.; Velardita, L.; Sandner, P. A decentralized application for the traceability process in the pharma industry. *Procedia Manuf.* **2020**, *42*, 362–369. [\[CrossRef\]](#)
22. Zheng, W.; Zheng, Z.; Dai, H.N.; Chen, X.; Zheng, P. XBlock-EOS: Extracting and exploring blockchain data from EOSIO. *Inf. Process. Manag.* **2021**, *58*, 102477. [\[CrossRef\]](#)
23. Nakamoto, S. Bitcoin: A peer-to-peer electronic cash system. *Decentralized Bus. Rev.* **2008**, 21260.
24. Vujčić, D.; Jagodić, D.; Randić, S. Blockchain technology, bitcoin, and Ethereum: A brief overview. In Proceedings of the 2018 17th International Symposium INFOTEH-JAHORINA (INFOTEH), East Sarajevo, Bosnia and Herzegovina, 21–23 March 2018; IEEE: New York, NY, USA, 2018; pp. 1–6.
25. Huang, Y.; Wang, H.; Wu, L.; Tyson, G.; Luo, X.; Zhang, R.; Liu, X.; Huang, G.; Jiang, X. Understanding (Mis)Behavior on the EOSIO Blockchain. *Proc. ACM Meas. Anal. Comput. Syst.* **2020**, *4*, 1–28. [\[CrossRef\]](#)

26. Nguyen, C.T.; Hoang, D.T.; Nguyen, D.N.; Niyato, D.; Nguyen, H.T.; Dutkiewicz, E. Proof-of-stake consensus mechanisms for future blockchain networks: Fundamentals, applications and opportunities. *IEEE Access* **2019**, *7*, 85727–85745. [[CrossRef](#)]
27. Sheikh, S.; Azmathullah, R.; Rizwan, F. Proof-of-work vs. proof-of-stake: A comparative analysis and an approach to blockchain consensus mechanism. *Int. J. Res. Appl. Sci. Eng. Technol.* **2018**, *6*, 786–791.
28. Liu, J.; Zheng, W.; Lu, D.; Wu, J.; Zheng, Z. Understanding the Decentralization of DPoS: Perspectives from Data-Driven Analysis on EOSIO. *arXiv* **2022**, arXiv:2201.06187.
29. Xu, B.; Luthra, D.; Cole, Z.; Blakely, N. EOS: An architectural, performance, and economic analysis. Retrieved June 2018, 11, 2019.
30. Bellare, M.; Yee, B. Forward-security in private-key cryptography. In Proceedings of the Topics in Cryptology—CT-RSA 2003: The Cryptographers’ Track at the RSA Conference 2003 San Francisco, CA, USA, 13–17 April 2003; Springer: Berlin/Heidelberg, Germany, 2003; pp. 1–18.
31. He, N.; Zhang, R.; Wu, L.; Wang, H.; Luo, X.; Guo, Y.; Yu, T.; Jiang, X. Security analysis of EOSIO smart contracts. *arXiv* **2020**, arXiv:2003.06568.

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.