POLITECNICO DI TORINO

Master's Degree Course in Computer Engineering

Master's Degree Thesis

Cybersecurity of devices and systems used in logistics and manufacturing

Supervisor
Prof. Guido Perboli

Co-supervisor
Prof. Stefano Musso

Candidate
Michele Carbone

April 2022
Torino

Abstract

Cybersecurity of devices and systems used in logistics and manufacturing

Context: The blockchain technology is a distributed database that records transactions between parties in a permanent and verifiable manner. For financial applications, blockchain has emerged as a leading technology layer.
In recent years, researchers and professionals' focus has switched to the use of Blockchain technologies in other fields becoming, among other uses, the backbone of a new supply chain.
However, the very nature of this Distributed Ledger Technology (DLT) raises some interesting issues around data privacy laws such as the european General Data Protection Regulation (GDPR).

Goal: This thesis has two main objectives, the first one is to analyze what are the main problems of incompatibility between blockchain and data privacy laws and how they can be solved, according to the main sources to date, with particular attention in case blockchains need to collaborate with IoT devices, which often have limited computing capabilities, now present in any field of major companies.

Finally, based on the study of the first part, a viable solution is proposed for the study case of the new Stellantis' blockchain under development for its Supply Chain and Logistics Networks in the production of batteries for electric vehicles.
This solution, applied to this blockchain recently in development, allows to comply with the privacy regulations of personal data, using lightweight systems and algorithms that can allow an IoT sensor mounted on the battery, called Controller, to safely participate as an actor in the blockchain, monitoring and updating the battery status until the final delivery of the vehicle.

Contents

List of Figures

Chapter 1

Blockchain and IoT devices

The objective of this first chapter is to provide a general overview of the concepts of blockchain and IoT.

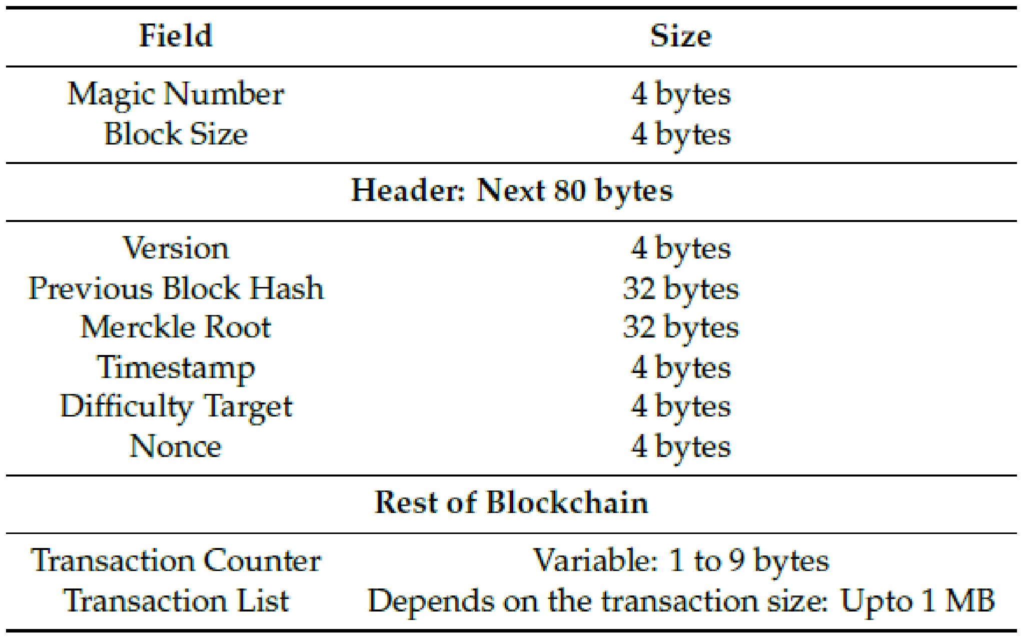
1.1 What is a blockchain?

The blockchain is a decentralized database that is shared among computer network nodes. A blockchain acts as a database, storing information in a digital format. This technology is well known for its use in cryptocurrency systems, for instance Bitcoin and Ethereum, and NFTs while it has found, in recent times, great uses in other fields among which it is becoming the base for a new supply chain. The blockchain's novelty is that it ensures the reliability and security of a data record while also generating trust without the requirement for a trusted third party.
The structure of the data on a blockchain differs from that of a traditional database. A blockchain organizes data into groupings called blocks, each of which contains a collection of data, called transactions. Blocks have specific storage capabilities, and when they're full, they're closed and linked to the preceding block, producing a data chain known as the blockchain (where the first block of the chain (block 0) is called **genesis**).
All additional information, added after that newly added block, are compiled into a new block, which will be added later to the chain, as the new last block.
A database organizes data into tables, whereas a blockchain organizes data into chunks blocks that are linked together, as the name suggests, creating a chain of blocks. When implemented in a decentralized manner, this data structure creates an irreversible data time line. When a block is filled, it becomes permanent and part of the timeline since when each block is added to the chain, it is given a specific time stamp [11].

In a blockchain network each transaction, before being added to a block, is validated by a community of nodes through a consensus algorithm.

1.1.1 Structure of a block in the blockchain

Each block is composed of two main parts: **Block Header** and **Block Body**.



Field	Size
Magic Number	4 bytes
Block Size	4 bytes
Header: Next 80 bytes	
Version	4 bytes
Previous Block Hash	32 bytes
Merckle Root	32 bytes
Timestamp	4 bytes
Difficulty Target	4 bytes
Nonce	4 bytes
Rest of Blockchain	
Transaction Counter	Variable: 1 to 9 bytes
Transaction List	Depends on the transaction size: Upto 1 MB

Figure 1.1: General structure of a block in the blockchain. Source: [6]

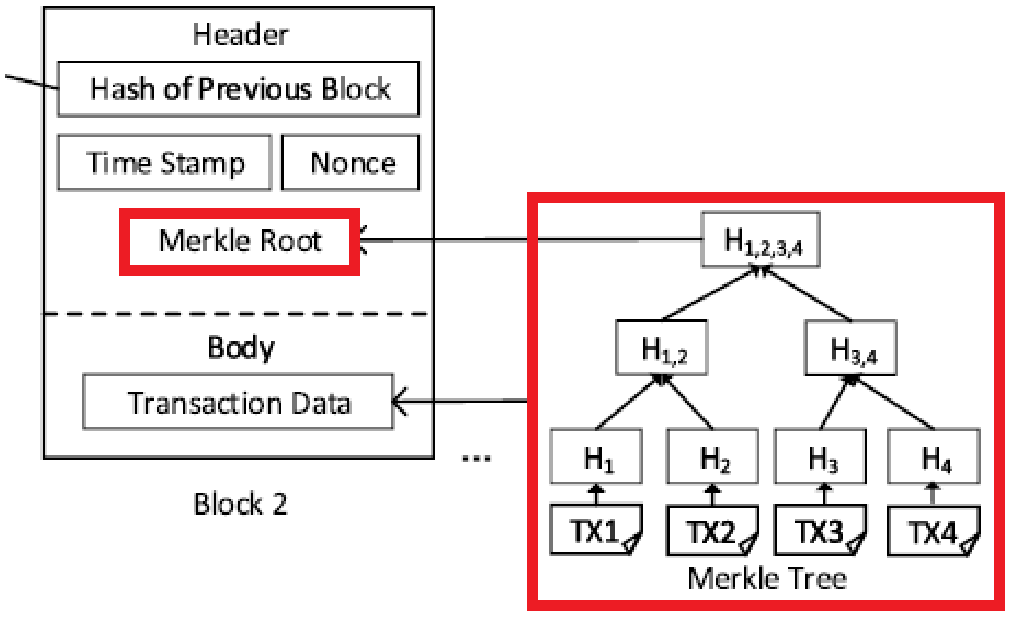
Header of a block

The header of a block contains various fields, among which ([24], [16]):

- The **hash of the header of the previous block** (0 if Genesis block).
- A **timestamp** that approximately indicates when the miner created the block. It's worth noting that block timestamps aren't perfect, and they don't have to be (they're only accurate up to an hour or two). Because a miner can falsify its own computer's clock, timestamps are not reliable for chronology.
- A **nonce**, which literally means "number used only once" is applied to blockchains that uses Proof of Work algorithm in order to reach consensus. In fact, it is the number that all miners strive to discover in order to validate a block and collect the mining bonus associated with it. When calculating

the hash, this nonce is used in conjunction with the block header: changing the nonce causes the hash to change.

- A **Difficulty Target** sets the maximum time a miner can take to add a new block to the network (e.g. when using Proof of Work, how hard it is to find a hash that will be lower than the target characterized by the system).
- A **version** field, indicates what version the blockchain was in during which that block was created, generally exploited to secure a backward-compatibility.
- A **Merkle root**: its the root of the merkle tree "stored" in the body of the block. As well explained by [16], the Merkle tree is a binary tree with each leaf node labelled with the hash of one transaction stored in the block body, and the non-leaf nodes labelled with the concatenation of the hash of its child nodes. Merkle root, i.e., the root hash of a Merkle tree, is used to reduce the efforts to verify the transactions in a block. Since a tiny change in one transaction can produce a significantly different Merkle root, the verification can be completed by simply comparing the Merkle root instead of verifying all the transactions in the block. The image below, from the paper [16], shows us the structure of a merkle tree and how this is connected to the merkle root and to the transactions in the body of the block.



Body of a block

The body of a block is the place where the actual transactions are stored. Each block may contain a transaction or a list of transactions. In the body we can find the transaction counter (number of transaction contained in that specific block) and the actual list of transactions.

1.2 Certification and validation of data entering the blockchain

Data entering a blockchain, e.g. coming from an IoT device, needs to be validated before a new block can be accepted, distributed and replicated by all nodes on the blockchain.
In general, there are two way to validate a block, the first is based on a consensus algorithms, while the second relies on a third party organization (outside the blockchain).

1.2.1 Data validation through consensus algorithms

According to [5], [15] and [26], every time a transaction is conducted on a blockchain, the transaction data will be stored in a new block. This new block will then be added to the blockchain. But before the block can be added to the chain, the information contained in it must be verified by the network. This happens by creating a so-called “hash”.
A hash is a uniquely identifier of the data in the block, since each data input, when hashed thorough an hash function, will lead to an unique hashed output (except for rare collisions) of a fixed size, such as 256-bit.

There are different strategies to elect the node that will create the hash value based on the consensus algorithm. This nodes on the network need to solve a complex “mathematical puzzle” and once the puzzle is solved, all other nodes on the network check if the calculations are correct.

The process of solving this puzzle and as a result creating a new hash is called “mining”. Most of the time, mining relies on sophisticated mathematical calculations and for this reason it necessitates a lot of computing power. It also necessitates the use of specialized computer hardware. As a result, not every network node will be able to act as a "miner."
To change any of the stored data in a block, a malicious actor would have to edit the

entire blockchain after that given block. However, because it takes far too much computational power, this is essentially impossible.

1.2.2 Miners earn a mining reward

Mining consumes energy, and energy costs money. Hence, there needs to be an incentive for miners to mine new blocks.
This incentive is called "mining reward" and is usually paid in the cryptocurrency native to the blockchain network, e.g. miners are rewarded with "ETH" cryptocurrency when mining blocks on a blockchain based on Ethereum.
There would be no new blocks if miners did not exist. As a result, the blockchain would be useless.

1.2.3 Ways to determine which node gets the mining reward

Multiple miners are competing for the mining reward at the same time.
Thus, blockchain networks need to apply a consensus principle that defines which miner will get the reward. There are different ways to accomplish this, among which, the two main ones are Proof-of-Work and Proof-of-Stake. However, recently there has been an increasing interest of companies in a third concept, called Proof-of-Authority.

Proof-of-Work (PoW)

PoW is the reward system commonly used in cryptocurrency networks such as Bitcoin and Ethereum.
In Proof of Work, an actor must solve a specific mathematical problem to be elected as a leader and determine the next block to be added to the blockchain. For example, let that mathematical problem be:

"*Given data X, find a number N such that the hash of N appended to X results is a number less than Y.*"

```
# Example - hash is a hypothetical hash function
# that has the outputs listed as below

Y = 10, X = 'test'
hash(X) = hash('test') = 0x0f = 15 > 10
hash(X+1) = hash('test1') = 0xff = 255 > 10
hash(X+2) = hash('test2') = 0x09 = 9 < 10 OK, Solved.
```

Given that the hash function in use is cryptographically safe, bruteforce is the only option to solve the challenge. To put it another way, the actor with the highest processing power is the one who will solve the aforementioned problem first the majority of the time, according to probability. These actors are also called miners.

The PoW has been widely successful primarily due to its following properties:

1. It is hard to find a solution for a given problem;
2. When a solution is found to that problem, it is easy to verify that it is the correct solution.

In Proof of Work, other nodes verify the validity of the block by checking that the hash of the data of the block is less than a preset number.
Due to the limited supply of computational power, miners are also incentivized not to cheat. Attacking the network would cost a lot because of the high cost of hardware, energy, and potential mining profits missed.

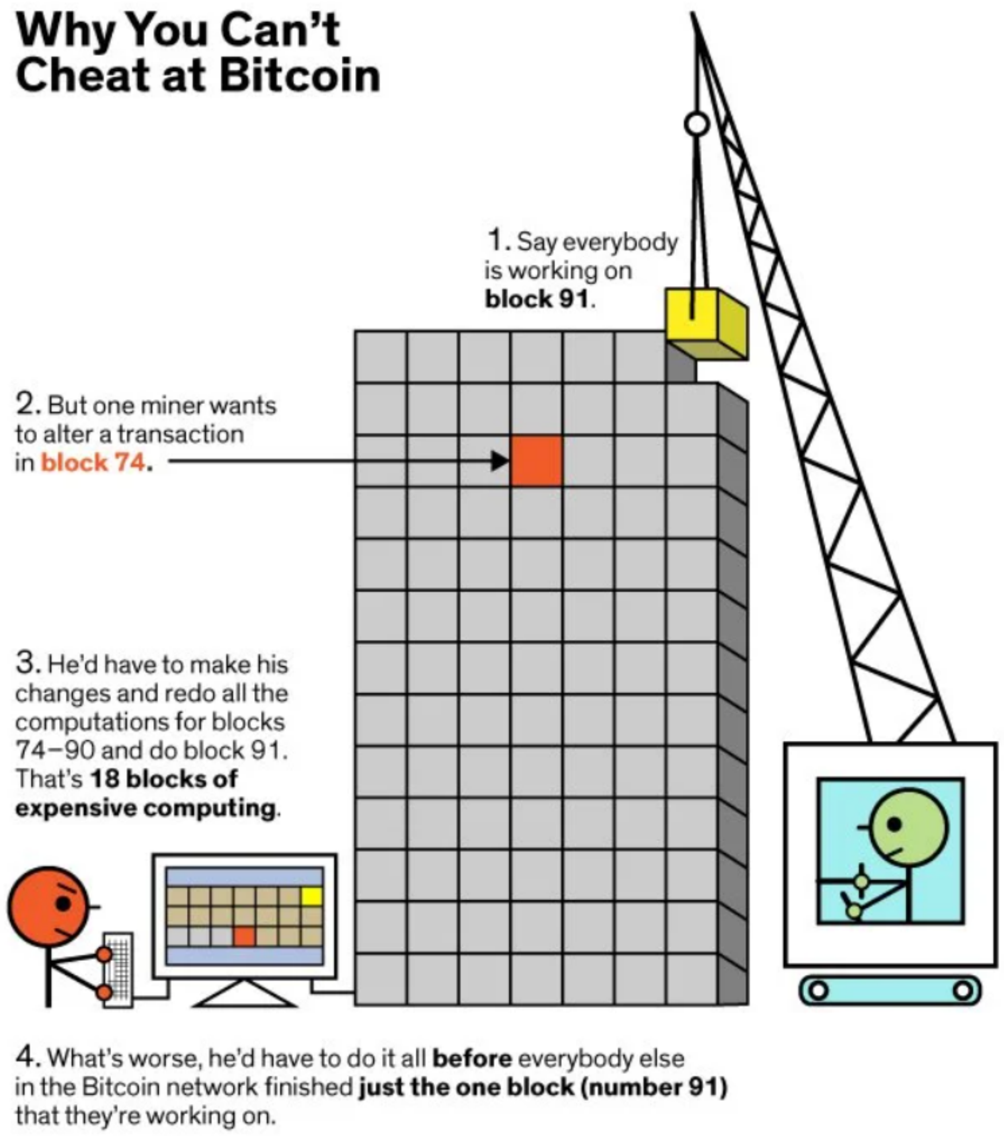

Why You Can't Cheat at Bitcoin
1. Say everybody is working on block 91.
2. But one miner wants to alter a transaction in block 74.
3. He'd have to make his changes and redo all the computations for blocks 74–90 and do block 91. That's 18 blocks of expensive computing.
4. What's worse, he'd have to do it all before everybody else in the Bitcoin network finished just the one block (number 91) that they're working on.

Figure 1.2: Image that shows why it is practically impossible to fight the integrity of a blockchain

Proof-of-Stake (PoS)

Loosely translated as "proof that you have an interest in the game, in the deal". PoS has the same goal as PoW: validating transactions by creating a new hash. However, in a PoS system, the nodes are not competing for the mining reward. Instead, a single node is selected to validate the next hash. The criterium for the

selection is the node's wealth – or in other words, it's stake in the network.
Thus, in a PoS-based network, the energy consumption will be much lower, because only one node is working on solving the mathematical problem (eg. it's estimated that both Bitcoin and Ethereum burn over $1 million worth of electricity and hardware costs per day as part of their consensus mechanism).
Moreover, in a PoS system, the reward is not paid in newly issued coins. Instead, the selected node will receive a transaction fee. All coins are already issued when the network is being created. That's why nodes who find a new hash in a PoS system are not called miners, but "forgers".
Stake is referred to as an amount of currency that an actor is willing to lock up for a certain amount of time. In return, they get a chance proportional to their stake to be the next leader and select the next block.

There are more ways to validate transactions, for example **Proof-of-Authority, Proof-of-Burn, Proof-of-Capacity or Proof-of-Elapsed Time**.
In principle, all these systems have the same goal: validating new data on the network. Only the way how the miners are selected will be different.

In Proof of Work, if Bob has more computational power and energy than Alice (and thus can output more work) he is more likely to win (mine the next block).

Similarly, yet again:

In Proof of Stake, if Bob has more stake than Alice, he is more likely to win ("mine" the next block).

From an algorithmic perspective, there are two major types of PoS: **chain-based proof of stake** and **BFT-style proof of stake**.

- In chain-based proof of stake, the algorithm selects a validator pseudo-randomly during each time slot (e.g., every 10 seconds) and grants that validator the right to create a single block, which must point to some previous block (normally the block at the end of the previous longest chain), and thus, over time, most blocks converge into a single constantly growing chain.

- In BFT-style proof of stake, validators are randomly assigned the right to propose blocks, but agreeing on which block is canonical is done through a multi-round process where every validator sends a "vote" for some specific block during each round, and at the end of the process all (honest and online) validators permanently agree on whether or not any given block is part of the chain. Note that blocks may still be chained together; the key difference is that consensus on a block can come within one block, and does not depend on the length or size of the chain after it.

Proof-of-Authority (PoA)

According to [1], although the PoW consent algorithm is the most reliable and secure currently existing, it has the big problem of not being scalable, leading to limited transaction-per-second (tps) performance. From this point of view, PoS consent algorithm's network have not yet managed to improve, having results similar to the PoW.
In this context, the Proof of Authority is currently being implemented as a more efficient alternative, as it is able to process a much larger number of transactions per second.

Proof of Authority (PoA) is a reputation-based consent algorithm, making use of identities. This means that block validators stake their reputation instead of coins. As a result, PoA blockchains are protected by validation nodes that are arbitrarily selected as reliable entities.
The Proof of Authority model is based on a limited number of validators, e.g. 4 up to 30 maximum in the IBFT2.0 by Hyperledger Besu, making it a highly scalable system. Blocks and transactions are verified by pre-approved participants, who act as system moderators.
This solution, makes blockchains more centralized, and for obvious reasons is not much appreciated by cryptocurrencies, but is extremely appreciated by large companies with logistical needs, and in reading this thesis you will notice how, not surprisingly, is the solution selected by Stellantis as consent algorithm for its blockchain.

1.2.4 Data validation through third parties

Before starting, just a little preamble about **permissioned** and **permissionless** blockchain by [23].
The basic distinction of these is clear from the terms itself. A permissioned blockchain needs prior approval before using it, whereas a permissionless blockchain lets anyone participate in the system.
Though the two systems might sound similar, they cannot be used for the same purposes. People might not be keen on using a permissioned cryptocurrency as one of the major drawbacks of cryptocurrencies is that no one has control over how it works.
For example, a company like Maersk, that uses blockchain technology to track its shipping logistics, will not want to store its confidential information onto a

permissionless blockchain.

Permissioned blockchains, also known as private blockchains, are closed ecosystems that can only be accessed by those who have been given permission to do so. Anyone who wants to validate transactions or examine data on the network must first obtain permission from a central authority. This is beneficial for businesses, banks, and other institutions that are confident in their ability to comply with regulations while also being worried about maintaining complete control over their data. A permissioned blockchain, such as Ripple, is an excellent example. The proposed solution for the Stellantis case, will be a permissioned blockchain.

Permissionless blockchains, also known as public blockchains, allow anyone to transact and join as a validator. The data on these blockchains is publicly available, and complete copies of the ledgers are stored across the globe. This is what makes it hard to censor or hack these systems. This blockchain does not have anyone who controls it, and one can remain relatively anonymous as there is no need for identifying themselves to get an address and perform transactions.

Summary:

- **Permissioned**:
  - Closed ecosystems (all partecipants are defined).
  - Only approved entities can run nodes.
  - Decentralization and transparency varies.
  - No anonymity.
  - Mining is not required (no tokens given to nodes as incentive).
  - Performance is drastically increased (scale too)
- **Permissionless**:
  - Anyone can access and create data.
  - Anyone can publish smart contracts.
  - Anyone can run a node.
  - 100% transparency.
  - Relatively high level of anonymity.
  - Performance is slow.
  - Scaling is challenging.

  - Monetary incentives to nodes (miners). Extremely energy inefficiant.

Having said that, according to [18] it is not always possible to validate data entering the blockchain without involving third parties.
Even if disintermediation allows increased speed of transactions and the cost reduction, the intermediaries who are being excluded from these processes may be in charge of valuable functions beyond simply recording a transaction.
Consequently, legal input is essential to understand what requirements must be fulfilled, and any regulatory frameworks (such as data protection and anti-money-laundering provisions) must be complied. For these reasons, blockchain members must be aware of the legal ramifications of the solution they are using, including public law, private law, criminal law and financial and regulatory law.
Concerning private law, the parties need to agree on applicable law, jurisdiction, general principles of proper governance, dispute resolution, privacy and the means of digital identity. Moreover, the identities of the parties must be established with sufficient level of certainty to validate the contract.
From a public law perspective there is the risk that permissionless blockchains are used for illegal purposes such as money-laundering or to take advantage of pseudonymous involvement to get around competition-law issues. Two main solutions have been analyzed to address the legal issues: **Combination** and **On-chain VS Off-chain**.

- The first one is based on a **combination** of permissioned and permissionless blockchains, where components of the proposed transactions require some intervention by a responsible party, such as compliance with specific regulations. All members and users of blockchains and smart contracts in which personal data are exchanged are data controllers and must comply independently with all data protection requirements, while all parties that run nodes in the blockchain are data processors and must comply with relevant provisions (more easy to manage in a permissioned than a permissionless blockchain).

- The second solution (On-chain VS Off-chain) consists in **separating what goes on the chain** or in the smart contract **and what is managed off-chain**. In this case, a real contract is stored off the chain, and linked to the chain through a hash secure value, so that the parties can have confidence that the agreed version is the one being relied on by taking advantage of blockchain's timestamping capability.

Before starting to use the blockchain framework all the parties involved in the logistics chain must agree on legal aspects such as the country jurisdiction they want to apply (in particular in case of international trades), dispute resolution, etc.

Furthermore, the data used to execute smart contracts must be accurate, and all parties involved in the chain must have faith in it. When data comes from IoT sensors, this is a critical consideration: the sensors must be validated (either by the manufacturer or by a third party), and all participants in the logistics chain must have faith in the data's accuracy.

1.3 What are smart contracts in the blockchain

Smart contracts are digital contracts, written in the form of a **simple program** stored on a blockchain, that are runned when predetermined conditions are met, as stated by [12]. This smart contracts are used to automate the execution of an agreement so that all participants can be immediately certain of the outcome, without any intermediary's involvement or time loss. They can also automate a workflow, triggering the next action when conditions are met.
As said just before, smart contracts are simple program, infact they follow the form of "if.. then.., else.. then..", so that the network of computers (for example IoT devices) will execute these actions when predetermined conditions have been met and verified. After the execution of a smart contract, the blockchain will be updated.
We will see in the following chapters that smart contracts can have various uses such as releasing funds to the appropriate parties, or most importantly for this thesis, make sure that the security measures against any cyber attacks are respected. To establish the terms of the smart contracts, participants must determine how transactions and their data are represented on the blockchain, agree on the rules that govern those transactions, explore all the possible exceptions, and define a framework for resolving disputes. Only after all of this, a smart contract can be programmed by a developer.
The benefits of using smart contracts are:

- **Speed, efficiency and accuracy**: Once a condition is met, the contract is executed immediately without the need of paperwork to be processed since smart contracts are digital and automated. This means that no time will be spent reconciling errors that often result from manually filling in documents.
- **Trust and transparency**: Because there's no third party involved, and because encrypted records of transactions, exploiting hash functions, are shared across participants, there's no need to question whether information has been altered for personal benefit.
- **Security**: hash functions are exploited, and since each block in the blockchain is connected with the precedent one, it is really difficult for an hacker to hack the whole distributed ledger.

- **Savings**: as a consequence of the three previous points, there is time and money saving since smart contracts remove the need for intermediaries to handle transactions and, by extension, their associated time delays and fees.

To give an additional context, in the environment of Ethereum blockchains, the most popular way to write smart contracts is by using Solidity, an object-oriented, high-level language specifically designed for implementing smart contracts.
This language is used to create from the most basic to the most complex smart contracts, having all the known constructors of any other programming language, starting from the most basic if/else, for/while loops going up to the most specific functions related to a blockchain allowing to transfer cryptocurrency, creating new contracts starting from another contract and so on.

1.4 What are IoT devices?

IoT (Internet of Things) devices are pieces of hardware, such as sensors, actuators, gadgets, appliances, or machines, that are programmed for certain applications and can transmit data over the internet or other networks, such as blockchains. They can be integrated into a variety of devices, including mobile phones, industrial machinery, environmental sensors, medical devices, and more.

1.4.1 Limitations and weaknesses of the IoT devices

In the challenge to be faced to improve cybersecurity in the network of IoT devices, some limitations of these devices must be taken into account, including:

- IoT devices typically have **low processing capabilities** and **limited memory**. The software developers are challenging themselves in order to design comprehensive security measures within a low memory of 64KB to 640KB. As a result, the CPU and memory are limited in IoT devices.

- Applying the updates to the IoT devices can be a challenge because some devices do not support updates and some older devices may not support new updates.

- **Power** is a critical aspect in IoT devices because their batteries are difficult to charge, and this might result in network failure due to a device's low energy. In the development of IoT devices and communication protocols, energy efficiency is a crucial concern. When the network is a blockchain, we strive to avoid using proof of work as an algorithm to gain consensus in the construction of new blocks since this type of algorithms consumes too much energy, as already stated in the precedent chapters.

1.5 Basic blockchain example in python

In this section, a practical solution implementing a basic blockchain on python, is proposed.

```
import hashlib


class MicheleCoinBlock:

    def __init__(self, previous_block_hash, transaction_list):
        self.previous_block_hash = previous_block_hash
        self.transaction_list = transaction_list

        self.block_data = "-".join(transaction_list) + "-"
        + previous_block_hash

        self.block_hash =
        hashlib.sha256(self.block_data.encode()).hexdigest()

t1 = "Gio␣sends␣2MC␣to␣Albert"
t2 = "Daniele␣sends␣3.1MC␣to␣Nicolet"
t3 = "Federico␣sends␣1.2MC␣to␣Mike"
t4 = "Mike␣sends␣0.4MC␣to␣Albert"
t5 = "Albert␣sends␣4MC␣to␣Federico"
t6 = "Nicolet␣sends␣1.2MC␣to␣Federico"

initial_block = MicheleCoinBlock("Initializer", [t1, t2])

print(initial_block.block_data)
print(initial_block.block_hash)

second_block =
MicheleCoinBlock(initial_block.block_hash, [t3, t4])

print(second_block.block_data)
print(second_block.block_hash)
```

```
third_block =
MicheleCoinBlock(second_block.block_hash, [t5, t6])

print(third_block.block_data)
print(third_block.block_hash)

#Simply giving a look at the code, we can see that in
#this blockchain, each block (MicheleCoin) contains
#two transactions, and any change to one of
#the block\_data will lead to the change of the
#block\_hash and all the next block\_hashes.
```

Chapter 2

Data protection regulations and cybersecurity within blockchain infrastructures

2.1 Privacy laws and blockchain

The very nature of **Distributed Ledger Technology (DLT)** raises some interesting issues around privacy laws and their application to DLT. It is important for industry participants to understand this important area of law and its application to their business.

Blockchains, just for how they are designed, are a technology that enables the secure validation, recording, and sharing of data. The data is stored in a distributed database, meaning there is not one centralised database controlled by a single person, but rather multiple copies of the database which are continuously updated in real time across the network of participants. Not only does this eliminate one single point of failure risk, but it also makes tampering with the data a significantly more onerous task, as a person would need to tamper with all copies of the data near simultaneously as stated by [14].

To safeguard data from unwanted access, blockchain uses cryptographic mechanisms. Cryptography is a type of data security mechanism that turns any type of data into a new format that can only be read by users who have been granted access to it, preventing it from being accessed by people without permission. Cryptography involves several steps to securely alter the format of the data. This is achieved through the following steps:

1. Converting the data into a "coded form", which is referred to as a hash, that

bears no resemblance to the original data, through an hash function;

2. Designing the hash such that it cannot be reverse-engineered, meaning it can only be decoded by guessing the underlying original data. A good hash function should be one-way (no way to obtain back an input from the output of the hash function), collision resistant and lead to a short output;

3. Storing the hashes in a manner which enables a user to easily confirm whether any of the original underlying data or hashes have been tampered with through the use of digital signature.

This enables a user to trust the integrity of the data once stored in the blockchain, without necessarily having to trust its counterparts and other users on the blockchain itself.

Blockchain technology exploits cryptography for wallets, transactions, security, and privacy-preserving protocols.

2.1.1 How does this structure tie into compliance with privacy law?

One of the key features of blockchain is its **purported immutability**, meaning that data stored in a blockchain cannot be subsequently altered.
The immutability of blockchain is often held out as **incompatible with data privacy laws**, such as the **General Data Protection Regulation "GDPR"**, that empower data subjects to have control over their personal data, including how it is collected and stored, and dictates that persons collecting and storing such data must agree to hand over, correct, and delete that data on request.
In addition to the data subject rights, the GDPR also contains a principle on **data minimisation**, whereby organisations should only process personal data that is relevant and necessary for the defined purpose, and the principle of **storage limitation**, whereby organisations should only keep personal data for as long as necessary for the purposes for which it was collected.

Apparently, these obligations are in conflict with the inherent "immutable" feature of the blockchain.
However, determining whether a specific network is compliant with the requirements of various data protection frameworks necessitates a case-by-case review. Furthermore, technological tools could be implemented into the blockchain network and related consensus protocol to make regulatory compliance easier.
In practice, many of the GDPR's privacy "challenges" can be solved by transparency, such as explicitly sharing with participants how the blockchain works and

utilizes personal data, as well as how the exercise of an individual's rights would be handled given the technology's nature. This necessitates a deep understanding of the nature of data processing activities by enterprises using blockchain technologies. The most common method is to contractually delegate the responsibility to notify the end-user to the party with the direct relationship.

2.2 Blockchain and the General Data Protection Regulation (GDPR)

Specifically, below are the themes that can allow a blockchain to cooperate with the European regulations of the GDPR (reccomended by [22]), which is claimed to be the **toughest privacy and security law in the world** according to several sources, including the European Union itself.
For this reason, working to meet the demands of this regulation could be enough to satisfy a comparison with any other regulation.

Can distributed ledgers be squared with European data protection law?
Many of the points of tension between blockchain and the GDPR are due to two overarching factors:

- First, the GDPR is based on the assumption that there is at least one natural or legal person – the data controller – to whom data subjects can address to assert their rights under EU data protection law in connection to each personal data point. These data controllers must adhere to the GDPR's requirements. Blockchains, on the other hand, are distributed databases that aim to achieve decentralization by replacing a single actor with a large number of participants. The absence of agreement on how to define (joint-) controllership impedes the distribution of responsibility and accountability.

- Second, the GDPR is predicated on the notion that data can be amended or deleted to meet legal obligations, such as Articles 16 and 17 of the GDPR. However, in order to preserve data integrity and enhance network trust, **blockchains make unilateral data alteration extremely difficult**. Furthermore, blockchains highlight the difficulties of conforming to data minimisation and purpose limitation criteria in the existing data economy.

The GDPR rule takes a binary approach to personal and non-personal data, and only the fist one is subject to its application. The Regulation expressly excludes anonymous data from its scope. In contrast to this legal perspective, reality functions on a spectrum that includes data that is clearly personal, data that is clearly anonymous, and everything in between.

Today, much economic value is derived from data that is not personal on its face but can be rendered personal if sufficient effort is put in place. Article 4 (1) GDPR underlines that personal data is data that directly or indirectly relates to an identified or identifiable natural person.

2.2.1 Principle of transparency

The principle of transparency requires that 'any information and communication relating to the processing of those personal data be easily accessible and easy to understand, and that clear and plain language be used'.
This implies that the information addressed to the data subject shall be:

- Concise
- Easily accessible and easy to understand
- Clear and plain language and, additionally, where appropriate, visualisation be used.

The necessary information can be provided in electronic form (such as through a website where it is addressed at the public).
Data subjects ought moreover to be made aware specifically of the 'risks' involved in processing.
Whereas the principle of transparency requires that the data subject be provided with specific information, there does not appear to be any need to inform a data subject of the specific technical infrastructure that is used to process their personal data. As such, **there is no requirement to inform data subjects that DLT would be used, only what personal data is processed and what risks arise**.
It is furthermore important to note that there is a link between transparency and the principle of purpose limitation.

2.2.2 Purpose limitation

Pursuant to the principle of purpose limitation enshrined in Article 5(1)(b) GDPR data shall be:

"collected for specified, explicit and legitimate purposes and not further processed in a manner that is incompatible with those purposes; further processing for archiving purposes in the public interest, scientific or historical research purposes or statistical purposes shall, in accordance with Article 89(1), not be considered to be incompatible with the initial purposes.".

This principle has two components:

1. Purpose specification
2. Compatible use.

Pursuant to the purpose specification principle, **personal data must only be collected for ’specified, explicit and legitimate purposes’** whereas the compatible use requirement mandates that personal data shall not be ’further processed in a manner that is incompatible with those purposes’.

In the case of blockchain technology, a specific question arises regarding the GDPR’s purpose limitation requirement, namely whether further processing of data added to blocks after the execution of the transaction for which it was originally added to the ledger is compatible with the purpose limitation principle. Because these databases are append-only, data will always be processed after it has been entered into the ledger.
For example, if personal data is used on a blockchain (whether in the form of the public key or transactional data) to execute a cryptoasset transaction, that data will continue to be processed even after that transaction has been successfully completed in the sense that it remains stored on the ledger, and continues to be processed according to the modalities of the used consensus algorithm.
From the perspective of the purpose limitation requirement, the question emerges whether the storage and subsequent involvement of such data in other transactions can be considered as part of the original purpose of processing, or whether it is necessarily incompatible with the purpose limitation principle. This issue will be addressed having regard to both the purpose specification principle and the compatible use requirement.

Blockchains and the purpose specification principle

According to Article 5(1)(b) of the GDPR, the purpose limitation principle requires that the controller communicate the purposes for which data is processed, and that the purpose be made explicit and legitimate. Purpose specification can be broken down into three distinct requirements:

1. The purpose of personal data processing ought to be **specified**, that is to say that it must be ’sufficiently defined to enable the implementation of any necessary data protection safeguards, and to delimit the scope of the processing operation’.
2. The purpose must also be **explicit**, meaning that the purpose ’must be sufficiently unambiguous and clearly expressed’.

3. The purpose of personal data processing ought to be **legitimate**. Here, the notion of legitimacy is not limited to the need for a legal ground for processing but rather requires that processing occurs in line with 'broader legal principles of applicable law' such as non-discrimination.

Data controllers relying on blockchain technology should thus clearly communicate to the data subject that they are using this technology and explain related implications such as that the processing is not limited to the original transaction but that their personal data will continue to be processed thereafter.
It is important to stress that while a disclosure along these lines could comply with the specificity and explicitness requirements, it wouldn't necessarily render the processing legitimate. Rather, a case-by-case analysis is needed to evaluate whether the fact that data continues to be processed past the initial transaction does not stand in the way of complying with other GDPR requirements (such as the right to erasure and the data minimisation requirement) and whether compliance with other applicable legal principles such as the non-discrimination requirement can be guaranteed. Where this is not the case, even a specific and explicit disclosure of the implications of using DLT for personal data processing would fall short of GDPR compliance.

2.2.3 Data minimisation

Pursuant to the principle of data minimisation, data ought to be

'*adequate, relevant and limited to what is necessary in relation to the purposes for which they are processed*'.

This means that only those data necessary for the controller's purpose are obtained and processed. This two characteristics of blockchains may be a cause of particular concern from a data minimisation perspective:

1. The ever-growing nature of such databases causes concern. Indeed, in distributed networks, data can only be removed or altered in extraordinary circumstances – meaning that obsolete data cannot be removed.

2. The replicated nature of data as in such distributed networks, each node stores (in principle) a full copy of the database, leading to the manifold replication of the relevant personal data.

This takes us back to the discussion regarding the appropriate interpretation of the purpose limitation principle and its application to distributed ledgers. Indeed,

if the purpose includes not only the initial transaction but also subsequent processing then arguably the replicated nature of these distributed databases and the continuous storage of data could be considered to be in line with purpose limitation.
This principle appears to be conventionally understood to relate to the quantity of data. From this perspective, blockchains do not appear as a technology that can easily be squared with data minimisation. An alternative interpretation would be that data minimisation is not so much about the quantity but rather the quality of data meaning that what would be required is that there is no processing of special categories of data unless absolutely necessary and that data is pseudonymised or even anonymised whenever possible.

2.2.4 Accuracy

The accuracy requirement stated by the Article 5(1)(d) GDPR mandates that personal data shall be

'*accurate and, where necessary, kept up to date; every reasonable step must be taken to ensure that personal data that are inaccurate, having regard to the purposes for which they are processed, are erased or rectified without delay*'.

2.2.5 Storage limitation

In accordance with the principle of storage limitation, personal data should be:

"*kept in a form which permits identification of data subjects for no longer than is necessary for the purposes for which the personal data are processed; personal-data may be stored for longer periods insofar as the personal data will be processed solely for archiving purposes in the public interest, scientific or historical research purposes or statistical purposes in accordance with Article 89(1) subject to implementation of the appropriate technical and organisational measures required by this Regulation in order to safeguard the rights and freedoms of the data subject*".

Article 5(1)(e) GDPR mandates that no obsolete data be retained. To ensure that personal data is not kept longer than necessary, '*time limits should be established by the controller for erasure or for a periodic review*'.
The storage limitation imperative raises the question of when data stored on DLT becomes obsolete, in line with the observations made above in relation to purpose limitation. This could be interpreted to relate to the completion of the relevant transaction, or it could be argued that even after this event, data is still 'necessary' for subsequent processing, in this case the continued storage of personal data on

the ledger as well as its processing in the context of the given consensus protocol. As amply discussed above, data can generally only be removed from blockchains in the most extraordinary circumstances, raising the question of whether the storage limitation can be respected in such environments.

2.2.6 Accountability

Article 5(2) GDPR mandates that the data controller is responsible for, and should be able to demonstrate compliance with, the requirements under Article 5 GDPR (examined just above). This relates to the duties of the controller, which have already been examined above.
The General Data Protection Regulation integrates accountability as a principle which requires that organisations put in place appropriate technical and organisational measures and be able to demonstrate what they did and its effectiveness when requested.
However, when we talk about implementing it in blockchains, as seen, these can be of different types, including permissionless and permissioned and obviously the latter is preferred, since it is possible to trace the identity of the real person behind a specific node of a blockchain, responsible for any non-compliance with one of the rules of the GDPR.

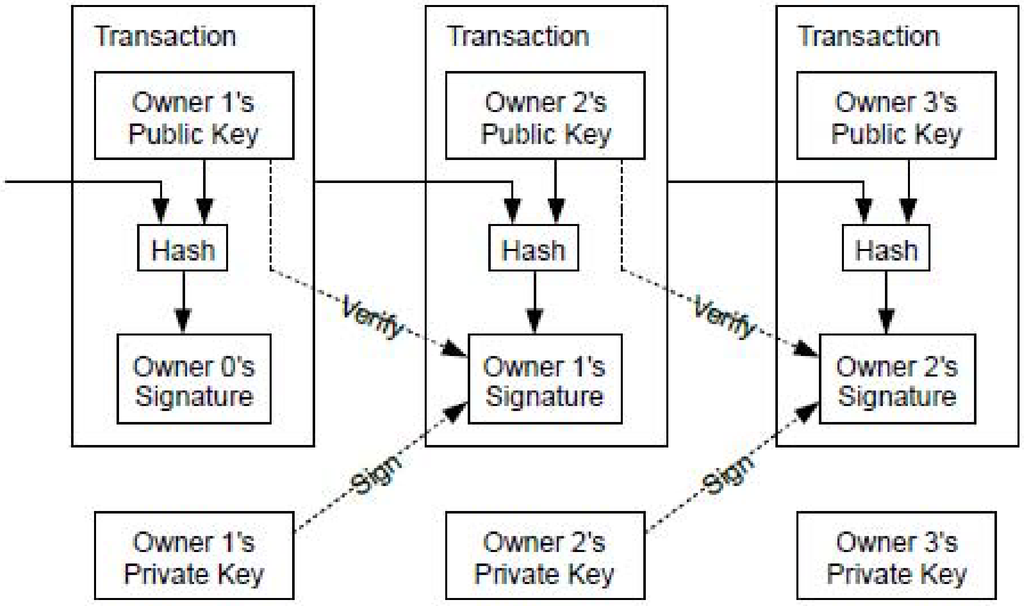
2.3 Cybersecurity in the blockchain

Quoting [10], the security of Blockchain technology is mainly based on three fundamental elements, known as the CIA's (Central Intelligence Agency) Triad:

- **Confidentiality**: the Blockchain offers extended functionality to ensure user anonymity. User keys are the only link between a user and their data. However, these keys are easy to anonymize.
- **Data Integrity**: Blockchains are designed as ledges where each block is linked to neighboring blocks using cryptographic hash functions. Therefore, once a transaction has been recorded on the blockchain, it cannot be changed or deleted. Any changes made to the data already recorded are processed as new transactions.
- **Availability**: having a large number of nodes guarantees the resilience of the blockchain even when some nodes are not available. And since each node on the network has a copy of the distributed ledger, the correct blockchain remains accessible to other peers even in the case of a compromised node.

2.3.1 Comparative Study of Blockchain Application and Security Issues

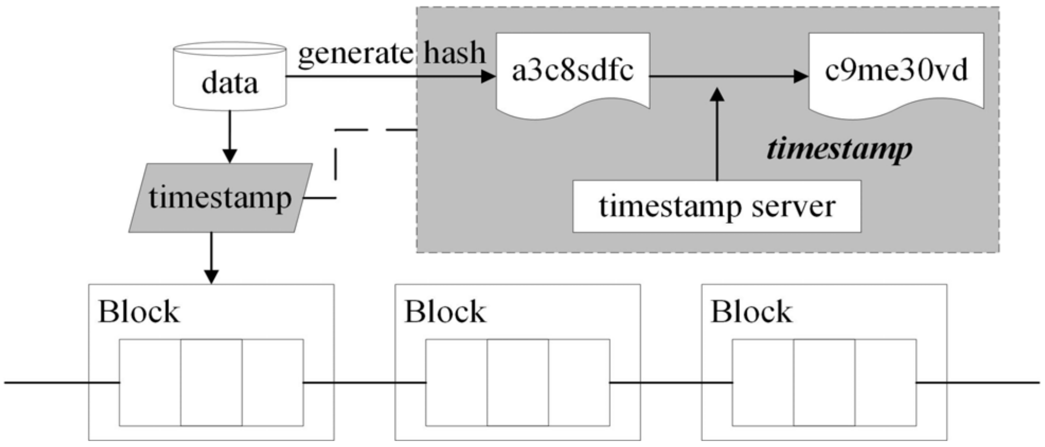
As already stated, in a blockchain system, data is generated and stored in units of blocks and consecutive blocks are connected in chronological order to form the chained data structure. All user nodes participate in the validation, storage and maintenance of data. Usually the creation of a new block should be approved by more than half of the users, and broadcasted to all user nodes to perform a network-wide synchronization. Once synchronized, the modify or delete operation is not allowed optionally [4].



Blockchain sacrifices proper computing capacity, bandwidth, electricity and storage resource for the improvement of security.

2.3.2 Tamper-proofing

Once a transaction is being created in the blockchain, a new timestamp will be recorded at the same time and any modification of data created before that timestamp will not be allowed any more.



Furthermore, a consensus method, as already stated in the precedent chapters, should decide if a new transaction can be recorded. To put it another way, writing data in blocks requires the consent of a certain number of users: typically, this percentage is set to greater than 50%.
To tamper with data during the data recording phase, adversaries must control more than half of the network nodes or have more computing power due to the consensus method.

2.3.3 Disaster Recovery

Unlike a traditional centralized database, which keeps data in one or more locations, every user in a blockchain-based application has the ability to generate data and preserve a full copy of it. This approach may result in some redundancy, but the network's stability and fault tolerance are improved since an arbitrary attack on one or more nodes will not cause the entire network to be destroyed.

2.3.4 Privacy Protection

Blockchain adopts asymmetric encryption mechanism to enable users to encrypt data with their own private key. Moreover, the hash value of a user's public key is calculated and perform as the ID indicator of the user. On one hand, the hash value has no relation with the real identity of user, thus keeping user's personal privacy information safe. On the other hand, the process of calculating hash value is not invertible, which means an adversary can't figure out a user's public key from the public user address, and calculating the private key from the public key is impossible. Therefore, blockchain achieves the goal of preserving user anonymity and privacy.

2.3.5 Technical Limitations

- The limited block capacity of blockchain inhibits its widespread use to a large extent. Initially, the capacity of a single block was set to 1MB to protect against DDoS attacks. And there has always been a debate over whether a larger or smaller block capacity is preferable. Because a larger block can hold more records, it meets the development criteria. However, larger blocks may make it more difficult to run and manage blockchain nodes. Smaller blocks, on the other hand, are easier to handle and more reliable, but space is exceedingly limited, particularly in complicated big data applications.

- Distributed storage mechanism creates a boarder attack surface in blockchain. A blockchain system chooses to store a complete copy of all data in every user's side. That means an attacker will have more alternatives to get access to those data. Although content in blockchain is not allowed to tamper, attackers can utilize other techniques such as data mining and correlation analysis to retrieve valuable information related to blockchain applications, users, network structure, etc.

- Consensus mechanism may trigger a cooperative attack. The consensus mechanism of blockchain is based on an assumption that the majority of nodes is to run and maintain the system. Once one or more nodes control more than 51% computing power of the whole system, they can join together to launch an attack to tamper with the content in blocks and conduct disruptive attacks such as DDoS.

Chapter 3

Practical applications to let blockchains and regulations cooperate

In the previous chapters was stressed as current blockchain protocols rely on a full copy of the ledger being replicated at least at each validating node to effectively validate the general state of the ledger. Some researchers, as stated by [13], consider this feature as profoundly at odds with some of the data protection principles laid out in **article 5 of the GDPR**:

- **Fair and lawful processing of personal data**.
- **Purpose specification**.
- **Adequacy**: collected personal data shall be adequate, relevant and not excessive in relation to the purpose or purposes for which they are processed.
- **Accuracy and retention**: personal data needs to be up-to-date and not retained longer than is necessary for the purpose it was obtained for. This two principles are linked to two of the rights introduced by GDPR: *the right to erasure and the right to amendment*, that allow data subjects to request the amendment or erasure of their personal data under certain circumstances.

About these two latter rights of erasure and amendment, blockchains by design retain the whole history of transactions as part of their strategy to guarantee decentralised validation. In the same tonic, it was also made very hard to delete or amend already committed transactions. Current blockchain protocols take advantage of the chain structure to optimize the way the ledger is constructed and validated, as such, it is extremely hard to delete or update a transaction in the chain.
The latter has been often marketed as a feature of data stored in blockchains to be

immutable. **Immutability is desirable in some cases, but, if in the context of personal data, it is in direct conflict with the rights of erasure and amendment**.

Below, some solutions are proposed by the paper [13], from the technical and legal point of view.

3.1 Proposed solutions from the technical side

Although most of the techniques that we are going to list have been developed for permissionless blockchains, they can be easily adapted to the permissioned ones. A large amount of work, from developers and technologists, especially in the context of permissionless blockchains, has been focused on the problems of

- How to avoid the identification of users from the analysis of blockchain transactions;
- How to obscure transaction contents from outsiders and validators, with a minimal tradeoff in the complexity of the validation protocol.

Indeed, the requirement imposed by current protocols of every validating node requiring to store a full copy of the whole ledger, gives the opportunity to attackers to join the blockchain as validators, download a copy of the blockchain, and apply data mining techniques to try to identify users. Permissioned blockchains are no exempt of this danger, as one party can use the same techniques to try to infer private information, like how many times other members of the network have transacted. It can be noted that most works has been targeting the preservation of the privacy of transactors, while the more **general problem of data protection has only been considered recently**.

Bitcoin developers suggest to always generate a new public/private key pair for each transaction, as reuse of the same creates patterns that are easier to detect by mining techniques. Other researchers have pointed out that analyzing the network's messages allows public keys to be linked to IP addresses, allowing for identification. Some experts believe that in the case of Bitcoin-type networks, the basic solution of hiding behind an anonymity network like TOR is insufficient.

Further efforts look at integrating advanced cryptography techniques into validation protocols to obscure potentially private information. For example:

- **Ring Signatures** is a technique that enable the signing of a transaction by several parties, in such a way that an outsider can determine that one of them generated the transaction, but is unable to tell which one did. By carefully selecting parties from the blockchain network to create a ‘ring’, a protocol can effectively protect the identity of the sender while still allowing validators to validate transactions.

- **Ring Confidential Transactions** go one step further and tackle the problem of obscuring the content of transactions, while still allowing validators to validate them. In this space, a combination of ring signatures with other cryptographic techniques was implemented (this was used in the Monero cryptocurrency until October 18, 2018).

- **Zero-knowledge proofs** is a general method by which one party (the prover) can prove to another party (the verifier) that he/she knows a value x, without conveying any information apart from the fact that he/she knows the value x. According to the European Parliamentary Research [22] Zero-knowledge proofs can be used to provide a binary true/false answer without providing access to the underlying data. **In the context of blockchains, this maps to the problem of a transaction sender wanting to prove to a validator that her transaction complies with the validation rules without revealing the values being transacted, or who is the recipient**. For example, succinct non-interactive zero-knowledge proofs (Zk-snarks) have been successfully implemented within the Z-Cash cryptocurrency.

- **Secure Multi-Party Computation (SMPC)** is a concept that the Enigma system leverages in order to achieve the same goal of zero-knowledge proofs but for Smart Contract platforms. SMPC is a subfield of cryptography that studies methods for parties to jointly compute a function over their inputs while keeping those inputs private.
  For example, a group of people could provide access to their salary, and together compute the average wage of the group without revealing the exact value of their salaries. Enigma combines a blockchain network for providing verification of hashes with a SMPC network where the data that correspond to hashes is split between different nodes, coordinated by a protocol that allows them to compute functions together without leaking information to other nodes. **Specifically, no single party ever has access to data in its entirety; instead, every party has a meaningless (i.e., seemingly random) piece of it**. Programmers can then specify which parts of the computation should be executed publicly (in the blockchain) and which in the SMPC, creating the **concept of Private Smart Contracts**.

3.2 Proposed solutions from the legal side

Despite technicians' activity in terms of strengthening the pseudonymisation capabilities of blockchain systems, there are very few technical papers specifically tailored to accommodate the **principles of erasure and minimisation**, perhaps due to the belief that if a high enough level of anonymisation of personal data within blockchain systems is achieved, the GDPR could be side-stepped from its very beginning. Lawyers picked up the gauntlet and have put forward some recommendations.
[8], for example, plays it safe and advocates to store personal data off-chain, having in the chain only a hash that could be used to link to an encrypted database (hash-pointer) where full data is stored. From the technical side, [9] identified the use of this technique in the wild, naming it "**Hashing-Out**".

- From a legal standpoint, hashing-out makes things lot easier, because when personal data is maintained in a database under the authority of an identified data controller, GDPR compliance is considerably easier. However, in some ways, this may be seen as a breach of the blockchain's decentralisation premise, as data control remains in the hands of a single centralised party. If a failure occurs (whether deliberate or not), data is irreversibly lost because it cannot be recovered from the hash. Furthermore, an availability failure (again, intentional or not) can block the entire data processing (single point of failure).
  Hashing-out is also an alternative for implementing the **right for erasure**. If a request to be forgotten is received, one only needs to erase any off-chain data that could be used to identify the subject if linked to the in-chain hash. A similar procedure can be used to implement the **right of amendment**: first, the incorrect record is 'erased', then, the amended data is added to the chain in a new transaction.

- **Chameleon hash** from the technical side, to devise **Redactable Blockchains**, has been explored. Informally, a **chameleon hash** is a hash that contains a digital 'trapdoor'. The knowledge of how to open the 'trapdoor' enables the 'breaking' of the hash. To amend a transaction, one goes to the block that contains it and uses the trapdoor to 'open' it, regenerate the block with the amendment and stitch it back to the chain in the original position. The knowledge of the trapdoor can be given to a trusted third party, or added as a primitive to the protocol to enable its decentralised execution. Note that, despite the fact that a party could try to redact a blockchain in its favour, it is still needed that all others accept the redacted version. Unfortunately, to be redactable a blockchain needs to include chameleon hashes since its inception making impossible to add redactability to existing Blockchains.

Lawyers have also noted that the right to erasure is not an absolute right, and that the concept of erasure leaves room for interpretation. This open the door to the following alternative solution: **when an erasure or amendment is requested, append to the blockchain a transaction that contains a reference to the one that is being erased/amended that semantically invalidates it**.

However, the applicability of such a solution depends on the significance of erroneous data being still visible, even if the blockchain attests its amendment. Consider the case where a blockchain is being used to store data regarding sexual offenders and a record of someone who has not committed such a crime accidentally appears in the blockchain. This citizen exercises his right to amend, and a transaction on the blockchain is pushed, effectively invalidating the record. Is this sufficient? Or does the fact that the incorrect record still exists pose a risk to the citizen?
Another situation can be a court to deny the "right to erasure" when the information is relevant for the public (public's right to access the historical record, with the potential impact on the person who asked for the right to erasure).
From a technical point of view, this ruling means that for the particular use case of using a blockchain to archive convictions, a hard delete functionality is required.

- **Erasure by encryption key destruction** is another possibility for erasure and relies on the use of a sufficiently strong encryption scheme for data stored in the blockchain. However, despite the mathematical proofs defining the limits of encryption schemes, the legal definition of what 'sufficiently encrypted' means is still under debate. Much will depend on how much of the dataset is encrypted; if only identifiers have been encrypted and some of the rest of the data is in clear, then reidentification is more likely via jigsaw ID techniques (the ability to identify someone by using two or more different pieces of information from two or more sources-especially when the person's identity is meant to be secret for legal reasons).

State-of-the-art encryption can ensure that data is safeguarded to a greater extent, i.e. that it is unintelligible to entities that do not have access to the decryption key, but it does not always imply anonymity. The possibility of identifying a data subject remains as long as the key or the original data are available (even in the case of a trustworthy third party contractually required to provide secure key escrow service).
However, the opinion also draws specific attention to the possibility of a brute force attack, which destruction of the key will not solve (although if the space of possibilities is so large as to suggest that a brute force attack is not a 'means reasonably likely to be used' by an intruder, this will boost the controller's defence). The controller may also have to keep the risk of reidentification under review

as technology evolves. For example, quantum computers could make a brute force attack possible. In my opinion, and in the opinion of the authors of the paper, current state of the art algorithms should be enough to protect against brute force, and even considering the possible realisation of quantum computers, the cryptography community is already developing post-quantum solutions.

3.3 Analysis and use cases of some solutions

3.3.1 Homomorphic encryption

According to [27], homomorphic encryption is a form of encryption that permits users to perform computations on its encrypted data without first decrypting it. These resulting computations are left in an encrypted form which, when decrypted, result in an identical output to that produced if the same operation had been performed on the unencrypted data. Homomorphic encryption can be used for privacy-preserving outsourced storage and computation. This allows data to be encrypted and out-sourced to commercial cloud environments for processing, all while encrypted.
According to the European Parliamentary Research Service [22], it has been argued that this could allow for the use of merely encrypted onchain data. Whereas, given the regulatory stance on encryption it is doubtful whether this would cross the GDPR's anonymisation threshold, **the solution could serve as one element in a broader anonymisation toolbox**.

3.3.2 Zero-knowledge proofs

As [20] pointed out, the worldwide data privacy landscape has recently been re-calibrating, with businesses under pressure from customers and regulatory constraints tightening following the implementation of the European's GDPR in 2018 or the California Consumer Protection Act, which went into effect in January 2019. To maintain regulatory compliance, protect company reputation, and ultimately remain competitive, companies must renew their data management procedures, taking use of the latest technological innovations.
Thankfully, in parallel to this regulatory shake-up, there has been sustained momentum behind the pursuit of Privacy-Enhancing Technology (PET) innovations. In particular, 2019 was a breakout year for zero-knowledge proof (**ZKP**) cryptography.

While blockchain has brought us great advantages like immutability and decentralisation, it can lack the privacy needed for some transactions. However, combining

zero-knowledge proofs (ZKP's) with blockchain technology has the power to provide users with a powerful mix of **immutability and security**.
As said in the sections above, zero-knowledge proof is an encryption scheme whereby one party (the prover) can prove the truth of specific information to another party (the verifier) without disclosing any additional information.
There are two main kinds of zero-knowledge proofs: **interactive** and **non-interactive** [7]:

- **Interactive** ZKP's involve a series of tasks or actions that the prover must complete to convince the verifier that they have particular information. Most of the required tasks undertaken in interactive ZKP's usually involve concepts of mathematical probability;

- **Non-interactive** ZKP's require no interaction between the prover and verifier, or the verification can take place at a later stage. These types of ZKP's require additional computers or software.

All zero-knowledge proofs include three essential prerequisites:

- **Completeness**: if a statement is true then the verifier can certify the prover possesses the required input.

- **Soundness**: the statement cannot be falsified, and the verifier cannot be convinced the prover has the required input when they do not.

- **Zero-knowledge**: the verifier will not be able to know any information beyond the statement being true or false. Details of the information and personal data of the other parties stay anonymous.

Like all forms of technology, zero-knowledge proofs have a range of advantages and disadvantages that can be listed below:

- **Advantages**
  - Simplicity: Does not require any complicated encryption methods.
  - Privacy: Increases the privacy of users by avoiding the reveal of personal information in public blockchains.
  - Security: Strengthens security of information by replacing ineffective authentication methods.
  - Scalability: Increases blockchain throughput and scalability.

- **Disadvantages**

- Limited: The protocols for ZKP's usually rely on mathematical equations and numerical answers. Any other method requires a translation.
- Requires a large amount of computing power: There are around 2000 computations per ZKP transaction that each require a certain amount of time to process.
- Restricted: If the originator of a transaction forgets their information, all the data associated with it is lost.
- Vulnerability: Potential vulnerability to advanced technologies like quantum computing.

According to the European Parliamentary Research Service[22], where zero knowledge proofs are used, the blockchain indeed only shows that a transaction has happened, not which public key (as sender) transferred what amount to the recipient. It has moreover been pointed out that **zero knowledge proofs and homomorphic encryption** have the potential to solve the conflict between data minimisation and the verifiability of data between many parties. A European Parliament report indeed appears to consider zk-SNARKs (Zero-Knowledge Succinct Non-Interactive Argument of Knowledge) as a means to comply with the data protection by design requirement.

While highly intricate, ZKPs can be tailored for easy deployment across a variety of business needs, and PET solutions such as these can help enterprises seamlessly transition into the compliance era. The list of associated use cases is growing, and industries such as identity and accreditation management, insurance and supply chain management are already benefiting from innovations in ZKP cryptography today.
Without compromising the privacy of important corporate information, ZKP cryptography can enable mutually beneficial collaboration between competitors, resulting in new income streams and efficiency advantages. ZKP encryption can provide a privacy layer that makes blockchain more accessible to businesses by resolving the data confidentiality issue that has impeded the adoption so far.

Although ZKT are state-of-the-art solutions and extremely interesting to use in the blockchain, in the specific case of the blockchain solution for the Stellantis supply chain, which is developed in the following chapters, it was decided not to use this type of technology, being computationally heavy and not recommended for small IoT devices, as we will see being the battery controllers in the specific case.

The discussion on ZKP techniques is concluded, listing below three possible uses of this in three different contexts.

Supply chain consortium

With an added privacy layer, a supply chain consortium can deploy blockchain technology to track assets along a supply route, without displaying sensitive transactional details that reveal confidential information pertaining to trading partners, sales volume, or pricing.

Messaging applications

End-to-end encryption has played a big part in allowing messages to be sent privately. However, traditional messaging applications require users to verify their identity to a server. With ZKP's, an individual can prove their identity without releasing additional personal information.

Complex documentation

Combining ZKP's and blockchain allows users to share complex documents with security. ZKP's have the potential to encrypt data in pieces, this enables users to control certain blocks and the visibility of the information contained within them, allowing some users access while restricting others.

3.3.3 Chameleon Hashes and Redactable (Editable) Blockchains

Before start talking about chamaleon hashes, let's see a little definition of what is a **hash collision**:

- A hash collision is a random match in hash values that occurs when a hashing algorithm produces the same hash value for two distinct pieces of data. Ideally, a good hash function should process the input value quickly while minimizing the possibility of collision, but the latter may exist.

Thanks to the information available in the papers [2] and [31], it can be noted that a **chameleon hash** is a cryptographic hash function that contains a trapdoor: without the trapdoor, it should be hard to find collisions, but knowledge of the trapdoor information allows collisions to be generated efficiently.
With the power to create hash collisions, any block can be replaced or even removed, making the blockchain effectively arbitrarily editable. In order to function, such a **redactable blockchain** network needs a trusted third party that is in possession of the trapdoor and can decide which block to edit or the trapdor can be added as a primitive to the protocol to enable its decentralised execution. Note

that, despite the fact that a party could try to redact a blockchain in its favour, it is still needed thatall others accept the redacted version. Redactions are published as chain updates, allowing arbitrary nodes to make a copy of the removed or edited entry before updating their chain.

An interesting article by University of Twente[21] shows an implementation of a chameleon-hash function with ephemeral trapdoor based on a chameleon-hash function able to redact single transactions without impacting the chain of blocks. To ensure that a single party does not have the power to compute collisions through the use of the trapdoor, they divided the secret into shares proposing the use of a weighted and verifiable secret sharing scheme. They described a redaction process that makes use of the standard transaction model of Hyperledger Fabric, and introduced the Proof-of-Redaction. This mechanism allows parties in the network to approve and reach consensus on a redaction proposal before the actual redaction.

The use of a Chameleon Hash function was considered and proposed for the development of the blockchain for the Stellantis' study case discussed in later chapters.

3.3.4 The addition of noise

Another possible solution consists in adding 'noise' to the data. Here, several transactions are grouped together so that from the outside it is impossible to discern the identity of the respective senders and recipients of a transaction. Algorithms similar to this model have already been defined for the Bitcoin and Ethereum blockchains. What is promising about this privacy technique is that the EU Article 29 Working Party (and Art. 68 of GDPR) has already recognised that, provided that the necessary safeguards are complied with, the addition of noise may be an acceptable anonymisation technique. For this to be the case, it should be combined with additional privacy mechanisms 'such as the removal of obvious attributes and quasi-identifiers'.

3.3.5 Scenarios

In this section, three scenarios by [13] are listed where a possible situation of real blockchain applications that must comply with the GDPR is staged.

Scenario 1

An individual interacts directly with a permissionless blockchain
For example, an individual that buys and exchange cryptocurrency without inter-

vention from a third party. From a role perspective, there is no escape from the fact that no data controller can be identified, making impossible to make someone accountable for any GDPR complaint. It is likely that the users will be held responsible for their own compliance, through terms of use that:

- Prohibit posting of certain kinds of personal data;
- Require users to have consent or another legal basis for processing.

Designers and developers of permissionless blockchains might consider to implement techniques described above, like zero-knowledge proofs, to offer at least partial guarantees to individuals. However, it has to be noted that, depending on the particular governance scheme used by the blockchain, changes that make the blockchain less compliant might be introduced.

Scenario 2

Applications that use permissionless blockchains as backend
In this case, a data subject interacts with an application that uses a permissionless blockchain as backend, e.g., a set of Ethereum Smart Contracts. Owners of the intermediate application are data controllers since they decide which personal data is gathered from the subject and what elements of it are stored and/or processed in the permissionless blockchain. As a result, they must warn the user that part of their personal information will be stored on a blockchain, as well as the mechanisms in place to hash, encrypt, and safeguard it.
Currently, the only way to guarantee GDPR compliance in this scenario is to hash out any personal data to a server controlled by someone that will be identified as a data controller. To decrease the possibility of pre-image attacks, state of the art salting techniques can be used. A salt is a random string that is concatenated to data to be encrypted, kept under the control of the data processor. Also, any table that matches pseudonyms (public keys) generated on behalf of data subjects required by the smart contract, to their identities needs also to be stored off-blockchain. In use cases where actual processing (instead of only storage) of personal data is encoded in a smart contract and a hash cannot be used, a possibility is the use of multiparty private computation schemes, however, although the encryption provided by the scheme might be considered enough for GDPR purposes, further research is needed to verify that the level of decentralisation offered by the additional network correspond to expectations.

Scenario 3

Permissioned blockchains
This case can be separated in further two, in a similar manner to the difference

between scenarios 1 and 2:

- The first subcase is an individual that joins on his own volition a consortium to run a permissioned blockchain. In this case, the individual needs to agree with the fact that others might process any data he inputs, and that he/she will be responsible of validating other's transactions, with the corresponding responsibility in case of a data breach or misuse. A possible role assignment in this case is that all members are joint data controllers. As such, they need to agree in a set of terms where they lay out how they are going to respond to requests of members related to any of the GDPR rights;
- The second subcase is when a permissioned blockchain consortium offers services to end-users, storing their personal data in their blockchain. Again, the simplest way to be in good terms with the GDPR a is that members of the consortium declare themselves as joint data controllers. In any case, the most important ingredient for compliance is common sense.

3.3.6 The Monero case

Monero is an open-source, privacy-oriented cryptocurrency that was launched in 2014.
Although it is often difficult to keep up with the technology used by the main blockchains, I found the Monero case very interesting, because it uses very advanced technologies to comply with privacy regulations. These technologies are also mentioned in the compliance document between gdpr and blockchain[22], although this document mentions that some of these technologies 'has proven to be highly porous and heuristic, with nothing even close to approaching high guarantees'.
To investigate this case, the online community suggest that the most up-to-date documentation is the code, closely followed by papers published by the **MRL** (Monero Research Lab).

Ring Signatures, Confidential Transactions and Stealth Addresses

As stated by [17], the the three key pieces of technology that Monero currently employs in order to achieve privacy are Ring Signatures, Confidential Transactions and Stealth Addresses.

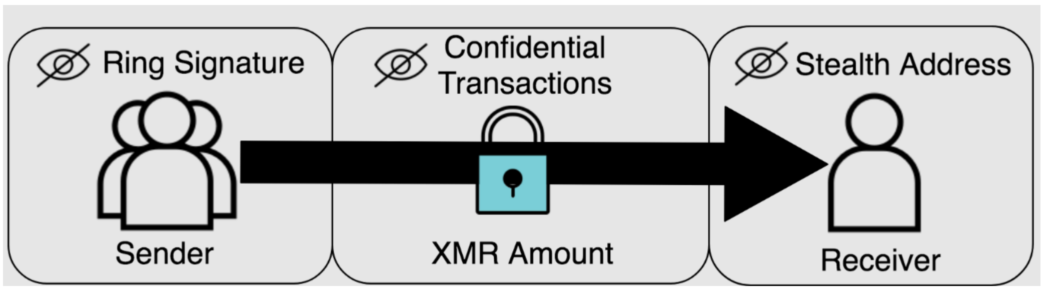


Figure 3.1: Ring signature, Confidential Transactions and Stealth Address in Monero Cryptocurrency. Image by [17]

The idea behind a Ring Signature scheme is simple, yet very powerful. The true sender of a message combines his or her own signature with multiple other signers to create a unified digital signature. Rather than a single identity, this unified digital signature represents a group.

The primary goal of a ring signature is to **enable the true signer of a message to claim plausible deniability, where each signer in a group has equal chances of being the real signer**. It's as if the police had a list of suspects that may have committed a crime, but no direct evidence that points to a specific person to even begin an interrogation.
The privacy of the real sender is achieved by pulling signers from past transactions in the blockchain and using their signatures as decoys. As a result, the real signer sending digital currency is mixed with signers of past transactions in the blockchain in an indistinguishable way:

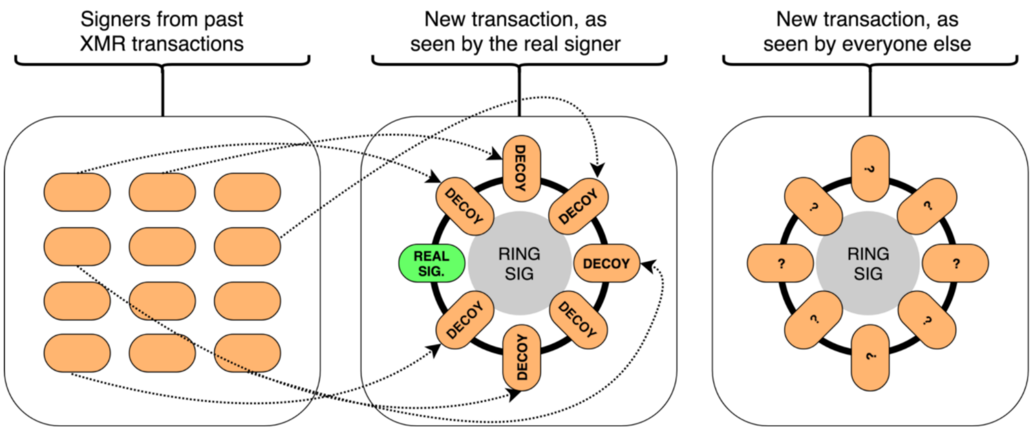

Signers from past XMR transactions
New transaction, as seen by the real signer
New transaction, as seen by everyone else
DECOY
REAL SIG.
RING SIG
?

Figure 3.2: Ring signature in Monero Cryptocurrency. Image by [17]

In Monero's terminology, a decoy signer pulled from historical transactions is called a **mixin**. The hypothetical transaction above has a mixin count of seven decoy signatures, in addition to the sender's real signature, and a total ringsize of eight signatures. **Note that this is a system of disassociation, where privacy is achieved by disassociating a single sender from a transaction. Sufficient privacy depends upon how many mixins a user decides to add to a transaction**. At the protocol's current iteration, there is a mandatory ringsize of 7, but users can decide to **increase** (and not decrease) ringsize as they wish (in the past there was the possibility to have less than 7 mixins, even 0 was permitted, however, as the Monero Research Labs originally found out, doing so hurts everyone else's privacy).

In addition to Ring Signatures, Monero also employs an encoding scheme called **Confidential Transactions (CT)** that hides transaction amounts. They call this combination **Ring Confidential Transaction** (**RingCT**). Despite popular belief, Confidential Transactions use **encoding** (which keeps data hidden, **immutable and verifiable**), instead of **encryption** (which keeps data hidden and **reversible**)

"RingCT Encoding != Encryption"

At a low level, the fundamental basis of Confidential Transactions is a cryptographic primitive for encoding called a **Pedersen Commitment**.
For context, cryptographic primitives are the building blocks of systems that use cryptography and are comprised of well-established algorithms like the SHA-256 hash function. The Pedersen commitment scheme used in RingCT has an additively **homomorphic property** [3.3.1], which, put (very) simply, allows multiple decoy inputs to be aggregated through addition.
A by-product of this process is **range proof** that proves that the amount committed by a given Pedersen Commitment falls within a certain range and is not a negative number.
The Confidential Transactions scheme also requires a special signature across all encoded commits within a transaction; a type of signature called a **Borromean Ring Signature**. What this means is that when a Monero wallet generates a Ring Confidential Transaction, not only is the signature of all ring members aggregated, but so is the amount of each input, which effectively **hides the transaction amount**.
For Monero, the activation of RingCT was one of its most significant updates to date. The adoption of RingCT has undoubtedly improved the way Monero wallets can source decoys because it eliminates the requirement of the value of each mixin input to be of a common denomination

Stealth Addresses

An interesting proposition from the CryptoNote white paper was the idea of a "wrapped" address to protect receivers, which Monero still uses. Rather than having the receiver's true address attached to an output and openly displayed, as is the case with Bitcoin, **the sender instead can create a temporary one-time address that can only be identified by the receiver**.
The term **Stealth Address** has been used to describe this mechanism and it provides a cleverly designed way to hide a transaction's destination. Before broadcasting an XMR payment, **the sender combines the receiver's public keys with a random number** in a key generating algorithm that creates a one-time key. The addition of randomness obfuscates the receiver's address, but the receiver can still identify it once the transaction has been sent to the network. Only the true receiver can do that by scanning the blockchain for a specific data point called the key image.
The one-time key generator referenced above is based on an Elliptic-curve Diffie-Hellman key exchange, which is a protocol where two parties agree on a key that unlocks a secret. In this case, the key image is an identifier that can only be located and spent by the intended receiver, which agreed on a common key with the sender. When a user sends XMR to the receiver, there is a single public key associated with that output and only the receiver can recreate its private key counterpart.

Since 2018 Monero began testing yet another highly sophisticated piece of cryptographic magic: **Bulletproofs**. This technology is intended to address one of the main drawbacks of RingCT: the size of the range proofs this scheme produces.

Bulletproofs

Click here for Stanford's site talking about bulletproofs. [25]
After working on the Confidential Transactions scheme, Monero teamed up with researchers from the Stanford Applied Cryptography Group to make it more efficient. Their research focused on applying a **non-interactive zero knowledge proof (NIZKP)** [3.3.2] system to aggregate all the range proofs of a Confidential Transaction and collectively prove their validity.
Relative to zk-SNARKs, the NIZKP system proposed by the Bulletproof white paper has both benefits and drawbacks. On one hand, the use of NIZKP Bulletproofs does not require a trusted setup for parameter generation, like Zcash's Powers of Tao ceremony. On the other hand, the verification of a Bulletproof is more time consuming than zk-SNARKs.

As discussed by MRL researcher Sarang Noether in December of 2017, under the current range proof format (in RingCT), the size of XMR transactions scales mostly linearly depending on the number of outputs (ex: 1 output = 7kB, 2 outputs = 13kB). Under bulletproofs, transaction sizes will then scale logarithmically instead (ex: 1 output = 2kB, 2 outputs = 2.5kB). **Therefore, this technology has the potential to greatly contribute to Monero's scalability**. The space savings granted by Bulletproofs may also enable the implementation of additional obfuscation mechanisms in future.

> "Bulletproofs are short non-interactive zero-knowledge proofs that require no trusted setup. A bulletproof can be used to convince a verifier that an encrypted plaintext is well formed. **For example, prove that an encrypted number is in a given range, without revealing anything else about the number**." [25]

3.3.7 Ring Signature implementation

An RSA-based ring signature technique, as well as one based on Rabin signatures, are described in the original publication. Solution details by [28] are listed below.

Signature generation

1. Calculate the key $k = H(message)$, using a cryptographic hash function, where "message" is the message to be signed;
2. Generate a random value "u";
3. Using a symmetric encryption function E_k, encrypt u, using the key k, obtaining $v = E_k(u)$;
4. Save the original v value on a support variable, such as $v_{supp} = v$
5. For each ring member but yourself, do:
   (a) Generate a random value x_i (x_s will be calculated using the signer's private key);
   (b) Calculate corresponding $y_i = g_i(x_i)$, where $g_i(x_i)$ is a trap-door function (i.e. an RSA public key in the case of RSA based ring signatures). So, for example $g_i(x_i) = {x_i}^{P_i} mod(N_i)$, where P_i is the public key of the member i of the ring (Ni is the product between the p and q choosen by the signer i while generating its key pair for the RSA algorithm). We are going to call $g_i(x_i)$ as "e";
   (c) Calculate $v = v \oplus e$

6. Solve the "ring equation" for y_s. This means that we have to force the actual value of v to be equal to the initial value of v (v_{supp}). This equation can be solved as long as at least one y_i and by extension x_i, can be freely chosen. This necessitates knowledge of at least one of the trap door functions' inverses (g^{-1}, knowing a private key), according to the RSA assumptions obtainig back an x_i starting from an y_i in this way: x_i = $g^{-1}(y_i)$.

   One way to do this in RSA is to calculate this: $x_s = (v \oplus u)^d mod N_s$, where d is secret key of the real final signer. The final ring signature is the (2n+1)-tuple formed by $\{P_1, P_2, ..., P_n, u, k, x_1, x_2, ..., x_n\}$, where the secret x_s will be mixed among the n-1 false secret keys.

Signature verification

1. Using the same symmetric encryption function E_k, encrypt u, using the key k, obtaining $v = E_k(u)$;

2. Apply the public key trap door on all $y_i = g(x_i)$, in practice:

   (a) Calculate $e = x_i^{P_i} mod N_i$;

   (b) Calculate $v = v \oplus e$.

3. If at the end $v = u$, the signature is valid.

Chapter 4

Blockchain, IoT and GDPR

Blockchain has been lately exploited in IoT to improve transparency, trust, and privacy. [30] is a really interesting book, taken as a reference from the blockchains' community, and has analyzed and listed a series of solutions and papers that allow a blockchain to cooperate respecting privacy regulations. Specifically, it was interesting for me to analyze, for the purpose of this thesis, the section of this book in which this research focused on the concomitant use of IoT devices in the blockchain, always respecting the privacy regulations.
For instance, a blockchain-based trust framework for collaborative IoT was proposed, where the framework by defining a set of smart contracts enforced IoT platforms to follow some specific rules in order to build trust in their communications.
Another solution, talking about a blockchain-based technique providing secure management of healthcare big data in IoT, was proposed. The technique combined the advantages of the private key, public key, and smart contracts to promote user privacy and make a patient-centric access control for electronic medical records.
Yet another one, describing a privacy-preserving blockchain based publish/subscribe model for IoT systems was proposed where the model protected data privacy and interests of subscribers and enabled publishers to control any data access.

Although aforementioned approaches took advantages of novel privacy-aware solutions (i.e., GDPR and blockchain) to safely protect user data, none of them proposed a combination approach to automatically verify GDPR rules over data processing units by monitoring a blockchain network. Moreover, **they did not consider the GDPR compliance of operations handled by IoT devices at design time before accessing or manipulating user data**.

Among all the solutions analyzed by the paper [30], two of them have particularly attracted my attention, as I found in the middle way between these two the

best ideas to think about and plan my proposed solution for the blockchain for the Stellantis' study case.
Below I have listed the general ideas behind this two papers, both showing application cases in IoT in the healthcare that respect the GDPR (according to [30]) using the blockchain, the first [3] uses privacy by design while the second [6] does not.

For a general context, medical care has become one of the most indispensable parts of human lives, leading to a dramatic increase in medical big data. To streamline the diagnosis and treatment process, healthcare professionals are now adopting Internet of Things (IoT)-based wearable technology. Recent years have witnessed billions of sensors, devices, and vehicles being connected through the Internet. One such technology—remote patient monitoring is common nowadays for the treatment and care of patients. However, these technologies also pose grave privacy risks and security concerns about the data transfer and the logging of data transactions. These security and privacy problems of medical data could result from a delay in treatment progress, even endangering the patient's life. The use of blockchain is proposed in order to provide secure management and analysis of healthcare big data.

4.1 Enhancing User Privacy in IoT: Integration of GDPR and Blockchain using privacy by Design

This document has been fundamental for me, since it allows to create a blockchain that respects "by design" the GDPR exploiting smart-contract, however it did not address in detail the technical aspects regarding the actual algorithms.
The paper [3] by representing an illustrative example shows how the integration of GDPR and blockchain can appear as **sub business processes** in the design patterns of IoT devices to protect EU citizens from privacy breaches. It **translates some GDPR rules into smart contracts** to facilitate the automatic verification of smart objects whose roles are data controller or processor. The paper also presents an **abstract model to demonstrate how users and IoT devices are connected to a blockchain network to access our GDPR-based smart contracts**.
The paper proposed four GDPR-based smart contracts and successfully designed a privacy-aware abstract model to improve the accountability of IoT devices, who are data controllers or processors of user data.

These smart-contracts are:

- **GDPR-Operation Contracts**: Containing four contracts, each one based on the four operations (access, store, profiling and transfer) that the actor (controllers/processors), can perform on user's data (the patient). This contracts assumes that the GDPR compliance is a boolean field, returned as a result by the smart contract itself along with the actor address and the processed personal data;

- **User consent contract**: This smart contract allows data subjects to vote and give a final consent in the form of a "yes" or "no" to the execution of operations that have already been claimed through the GDPR-operation contract;

- **Submission Contract**: Smart contract submission is ensured through this smart contract, which will collect the used personal data, operation and adresses, and store them in a log which will then fulfill a new transaction in the blockchain;

- **Verification contract**: An authorized third party verifier will be able to easily verify that all the operation done in the blockchain are compliant with the privacy regulation through this smart contract which is able to retrieve all actors, operations and personal data, verify them and report violations. Eventually the verifier is able to cast a vote (which state if there is actually a real violations) that will be stored in the blockchain. Any violations is reported to the verifier in 72h, as required and stated by Art. 33 of GDPR.

Results obtained with this solution

This paper proposed a blockchain-based approach, supporting GDPR, to protect the personal data of IoT users. They expressed the purposes of data processing of IoT devices through some typical operations. A set of GDPR rules in the form of legal questions were assigned to each operation. Such operations along with their relevant legal questions facilitated the translation of GDPR rules to appear as opcodes in smart contracts. The paper proposed some GDPR-based smart contracts and designed a privacy-aware abstract model to improve the accountability of IoT devices, who are data controllers or processors of user data. Furthermore, **several design patterns were presented to show how the right to data privacy with the aid of blockchain and GDPR can be added to the business processes of IoT devices at design time**. Such patterns enforced devices to carry out any operation on personal data under the user consent. Otherwise, they were classified to be violators.
Eventually, the proposed smart contracts were deployed in Ropsten test network and the results indicated that there is a direct relation between the number of operations executed on personal data and the cost spent on the verification of operations in

accordance with GDPR rules. The mining time was, however, independent of the complexity of proposed smart contracts.

4.2 A Decentralized Privacy-Preserving Healthcare Blockchain for IoT (not privacy by Design)

Unlike the precedent paper, this one was extremely interesting because, taking advantage of some of the algorithmic concepts we talked about in the previous chapters, it proposes a solution that allows you to comply with privacy regulations. In this case the solution is not "by design", as it is more from the algorithmic point of view, and is therefore feasible later in any blockchain.

The paper [6] starts by reminding us that blockchains are computationally expensive, demand high bandwidth and extra computational power, and are therefore not completely suitable for most resource-constrained IoT devices. In this work they try to resolve the above-mentioned issues of using blockchain with IoT devices proposing a novel framework of modified blockchain models suitable for IoT devices that rely on their distributed nature and other advanced cryptographic primitives. The solutions given here make IoT application data and transactions more secure and anonymous over a blockchain-based network.

In this paper, blockchain is the adopted solution for data privacy and security since it provides the robustness against failure and data exposure. However, adopting blockchain in the context of IoT is not straightforward and entails several problems such as demands of high computational power to solve PoW, low scalability and long latency for transaction confirmation over the network. To deal with this, they propose a novel model of blockchain and eliminate the concept of PoW to make it suitable for IoT devices, relying on the distributed nature and other additional security properties to the network. Transactions in blockchain are broadcasted publicly in the blockchain network and contain additional information about both the sender and recipient.

- A lightweighted Ring signature, which we talked about a few chapters above, is adopted for anonymity and authenticity of the user preserving his privacy (in this paper they are not using heavy operations such as pairing and exponentiation in the ring signature to make it more suitable for blockchain and IoT).

- A lightweight encryption algorithms (ARX ciphers, which only uses simple arithmetic operations, namely modular addition, bitwise rotation and XOR) and public encryption schemes are used to perform double encryption of data. They firstly encrypt the data using symmetric key encryption and then encrypt the symmetric key itself using a public key (so they are not encrypting the same data twice).

- Diffie–Hellman key exchange technique is used to securely exchange cryptographic keys over a public channel.

Using these techniques together they, guarantee the security, privacy and anonymity of user's data using lightweight techniques suitable for small IoT devices.

The main concern in Remote patient monitoring systems is the secure and efficient transmission of the medical data since this data are a lucrative target for hackers. Blockchain technology in its original form is not enough, so some modifications are required. The paper proposes some solutions, infact in their model they are using:

- **Decentralization**: through decentralized network obtaining robustness and scalability.
- **Authentication of data**: through lightweight digital signature they are ensuring that data is not tampered nor modified or lost during the transmission.
- **Scalability**: through the elimination of the concept of PoW and by dividing the overlay network into several clusters instead of a single chain of blocks. In this way a single blockchain is not responsible for all nodes, since these are spread over several clusters.
- **Data Storage**: through cloud servers storing encrypted data blocks, since storing IoT big data over blockchain is not practical. Data in the cloud servers is safe due to additional cryptographic security systems like digital signature and other high standard encryptions.
- **Anonymity of users**: through lightweight Ring signature obtaining anonymization of sensitive informations over the network.
- **Security of data**: through double encryption by encryption of the data (through ARX algorithm) and then encryption of key which was used to encrypt data (using the public key of the receiver). In addition to this, Diffie–Hellman key exchange technique is used to transfer the public keys and therefore making it almost impossible for the hacker to find the used key.

4.2.1 Implementation, security evaluation and conclusion

The paper [6] continues by showing a final overall look to the model, remembering that there is a patient which is equipped with a wearable device (such as a blood pressure monitor) that after registering to the network (by proving all the identity veirification documents), can connect to it. The health informations gathered by the wearable device are sent to a smart device (e.g. a smartphone) for the formatting and aggregation. After that, this formatted data are sent to the competent smart contracts that check that the values respect the threshold ones. If they are not respected, the smart contract will act in this way:

- An event will be created by the smart contract itself and an alert will be sent both to the overlay network and the patient.
- Add a digital signature to the abnormal readings, and store them in the cloud servers, which will then transfer the hash the data received from the smart contract to the overlay network.
- Finally, an Health Alert Event will be sent to the overlay network (pay attention, the health readings are **not** stored in the overlay network: just the alert will be stored). This health alert event will be anonymous since privacy is preserved thanks to the advanced cryptographic techniques we talked about in the above section, where the "sender" is the smart contract and the "receiver" is the overlay network itself. The concept to have in mind is that the overlay network contains all the public key information of all connected nodes and hash indexes of the stored data in the cloud server.

After receiving the hash of the data and the Health Alert Event, the overlay network will send the latter to the healthcare provider which will be able to access the full health reading of the patient for which he/she is authorized over the network. A summary of this is provided by the paper itself below.

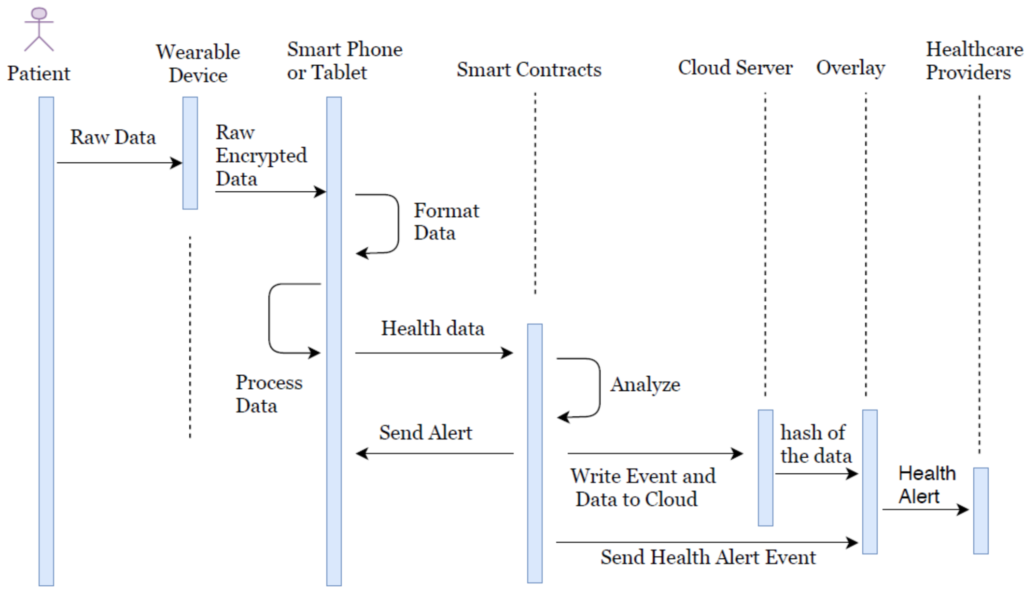


Figure 4.1: Logical flow execution of the system by [6]

In conclusion, the architecture solution, proposed by the paper [6], leveraging the blockchain successfully solves most of the security and privacy threats while considering the resource constraint factor of IoT.

Chapter 5

Blockchain in Supply Chain and Logistics Networks: the Stellantis' study case

This chapter aims to propose a possible solution that allows compliance between the blockchain under development by Stellantis and the personal data regulations. The purpose of this blockchain is to monitor all stages in the supply chain of battery packs assembly for electric vehicles.

This work, done in collaboration with the CRF (Centro Ricerche Fiat), is part of an initial preparatory research for a European project to which the "Politecnico di Torino" has recently begun to participate.
This project, called Concordia Horizon 2020 (Concordia H2020), is one of the four projects launched by Horizon 2020 and aims to establish a user-centric EU-integrated cyber security ecosystem for digital sovereignty in Europe.

The following chapters will show the proposed solution, based on two pillars:

- The first focuses on the use of smart contracts that will automatically check the compliance with the requirements of the GDPR;
- The second, on the other hand, deals with the algorithmic and cryptographic aspect, proposing light and safe solutions good for IoT devices, such as controllers, which have limited computing power.

Finally, to further satisfy the requests of the GDPR, solutions will be proposed that focus on:

- The implementation of clear and transparent communications to the data subject.

- The use of chameleon hash as the hash function used in the blockchain itself, thus allowing the cancellation and / or modification of blocks in the blockchain.

5.1 Scenario: battery assembly in electric vehicles

5.1.1 Proposed workflow of the actual blockchain

[19] proposes a workflow where Stellantis:

- Assembles a battery pack containing a controller;
- A new digital battery resource is created in Blockchain and associated with the controller digital identity;
- Only the controller can update the digital battery resource;
- Physical operations on the battery pack must be allowed by the controller;
- Based on the battery pack position, the controller will check a different policy in order to allow a physical operation;
- Physical operations on vehicles undergo the same protocol;
- Eventually, when the dealer receives the vehicle, the monitoring can be stopped.

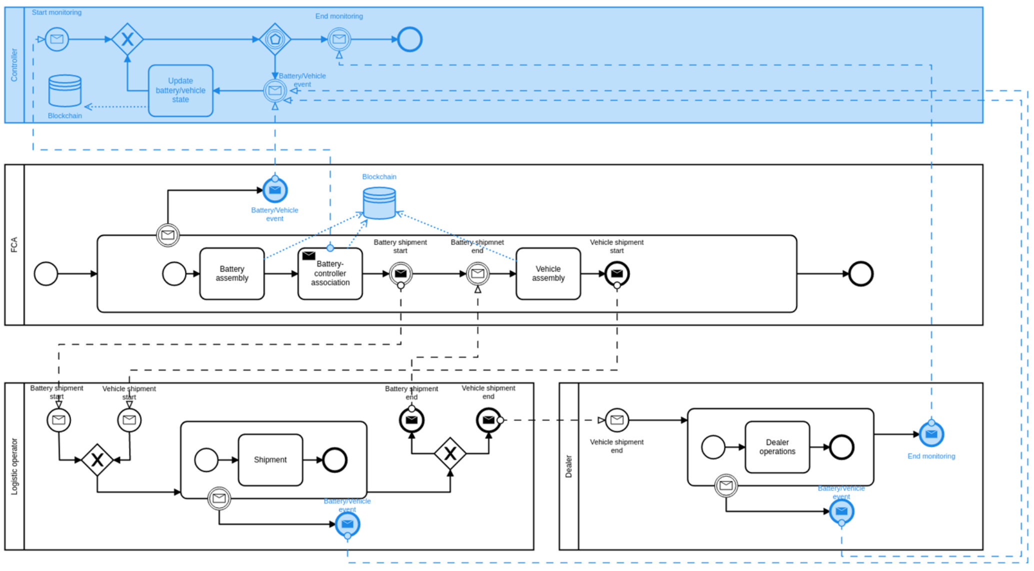


Figure 5.1: BPMN by [19].

Note: due to the latest changes, a requirement is that data incoming to the blockchain, are not coming directly from sensors, but from a Stellantis central database (resulting in a modification of their BPMN, present above).

Even if this goes against the concept of decentralization, which makes blockchains strong, the goal of this first phase of study by Stellantis, is to have a blockchain, that in hybrid mode with their central database can allow to monitor their supply chain section regarding the batteries of the electric vehicles.

The blockchain they are working on, is based on a centralized **Hyperledger Besu**, an Ethereum client setted on a permissioned mode and is planned to have up to 10 nodes. The choice for a centralized blockchain is common among hihgly regulated industries, where non-compliance with the rules, such as the insertion of malicious blocks, is extremely unlikely since different nodes are often one representing different partners working towards the same goal. In the rare case where a malicious event happens, this can be punished through more practical methodologies, such as fines to the entity representing the malevolent node.
As consent algorithm, it is used **IBFT 2.0**, that is a proof of authority consensus protocols which is perfect in such an environment, where there is mutual trust between the participants, so between the nodes of the blockchain.
Proof of authority consensus protocols have faster block times and a much greater transaction throughput, which is perfect in a blockchain that aims to implement IoT devices in the imminent future. When implementing IBFT 2.0 there are no forks to the blockchain (all blocks are included in the main chain), obtaining what can in fact be called an extremely enhanced and distributed database.

Goal of this section of the thesis, is to make this blockchain one that operates in compliance with data regulations. Below there is a solution proposal that is allowing us to achieve this, using smart contracts and safe and lightweighted cryptographic algorithms.

5.2 The start

In order to make the proposed blockchain, one that is regulation compliant, it was deemed appropriate to take as reference two papers (of which I talked about in previuos chapters), that I found fundamental in my study for this thesis:

- Enhancing User Privacy in IoT: Integration of GDPR and Blockchain, pages 322–335. By Masoud Barati and Omer Rana. I will refer to this paper as [3];

- A decentralized privacy-preserving healthcare blockchain for IoT Sensors. By shutosh Dhar Dwivedi, Gautam Srivastava, Shalini Dhar, and Rajani Singh. I will refer to this paper as [6].

5.2.1 System scenario

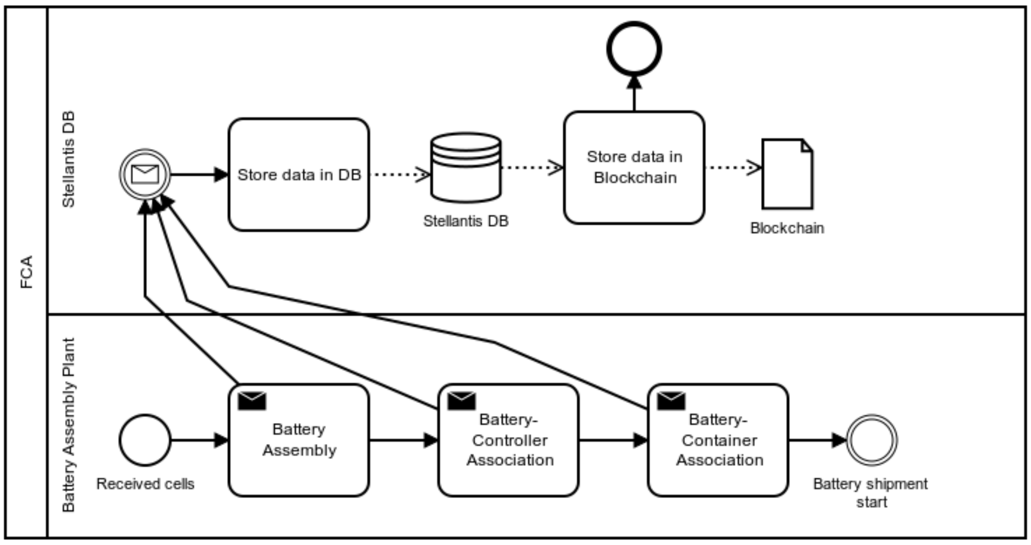
The battery assemblement

To better understand the whole scenario, the reader is advised to keep the BPMN as a constant reference (figure 5.1.1).

Battery cells, coming from a cells production plant, are assembled in a Battery Assembly Plant (namely in Turin Mirafiori, FCA) in order to create new battery packs, each one with a unique digital identity (namely BatteryID), which is recorder in the Stellantis DB. This DB will then create a new event in the blockchain, recording the new battery ID.
During this phase, a controller (an IoT sensor with a unique digital identity, namely ControllerID) is mounted to the battery and linked to it through the blockchain with a procedure similar to the previous one: the BatteryID and ControllerID will be linked in the Stellantis DB, that consequently will create a new event in the blockchain.

The last operation to be performed by the operator in the Battery Assembly Plant, is to put the new battery pack in a specific container. This container, with an unique identifier (namely ContainerID) will be linked to the battery it contains in the Stellantis DB, which once again will create a new event in the blockchain.

At this time a partial BPMN can be obtained, superimposable to the main one (figure 5.1.1).



Whenever a new controller is produced by the Stellantis System, there will be two main operation useful for the IT security of subsequent operations:

- The controller is able to generate a public-private key pair, storing secretely its private key, and sharing the public key to the central Stellantis' DB, which will save it for future uses;
- Eventually, the Stellantis' DB will share its public key that will be stored inside the new Controller.

The start of battery shipment

The container is now stored in the Intermediate Warehouse in "Orbassano, Arcese", ready to be shipped to the Vehicle Assembly Plant (namely in Melfi, FCA). While the batteries are stored in the warehouse, and while the latter are traveling on the rail transportation towards the Vehicle Assembly Plant, local operators at Arcese level are entitled to operate on the battery packs, whether by recharging or by isolating the packs in case of abnormal behavior.
During this phase, any operation on the battery will be recorded by the Controller on the Stellantis DB, which will then create an event linked to the battery in the Blockchain. At the same time, **Container Events** will be recorded on the same blockchain. This events, are used to update the actual container position, in such a way to give a position context to all the other battery events and answer to the question "Where was the battery pack when this event happened?".

According to the documentation, the Container Event can generate one transaction per event. This transaction has the following form: ***Container position***

: "Message", where the container position can be thought as the actual GPS position of the container, and the message can be one of the following:

- Full container
- Departure from supplier
- Arrival at CDC
- Departure from CDC
- Arrival at advanced warehouse
- Departure from advanced warehouse
- Arrival at assembly plant
- Departure of empty container

Arrival at the Vehicle Assembly Plant

Once the container arrives at the Vehicle Assembly Plant, the battery is removed from the container which is then empty and ready to be shipped back to the Battery Assembly Plant (a last Container Event related to the battery is generated: ***Container position* : "Departure of empty container"***).

The battery pack is now assembled in the vehicle and tested in the production lines. Starting from this moment, the position context of the battery is no more given by Container Events, but by a new type of events called **Vehicle Events**. After the final test, the battery is recharged by the internal FCA operators and the final vehicle is stored on the car lot: the vehicle shipment starts from this point. During this phase, FCA or i-FAST operators remotely monitor the vehicle charge and operate on the vehicle to re-charge in case of need. As always, any action performed on the battery is captured by the controller, recorded on the Stellantis DB, which then create a new transaction on the blockchain.

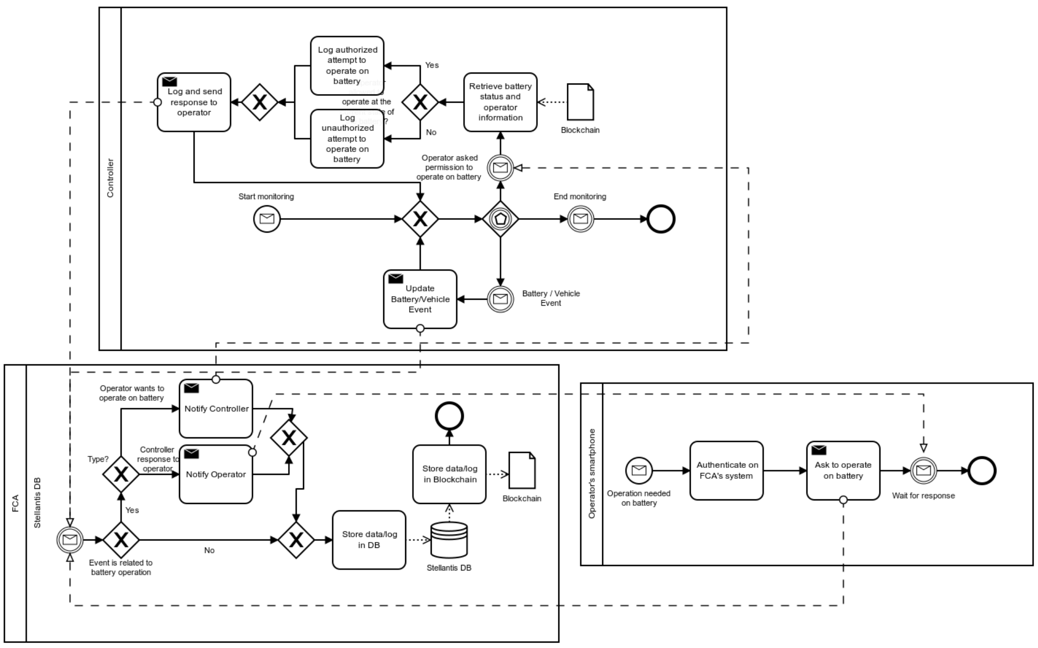
According to the documentation, the Vehicle Event can generate one transaction per event. This transaction has the following form: ***Vehicle position* : "Message"***, where the vehicle position can be thought as the actual GPS position of the vehicle, and the message can be one of the following:

- Assembled vehicle
- Departure from assembly plant

- Arrival at interport. (**Note**: (Melfi car lot))
- Departure from interport
- Arrival at harbour 1 (**Note**: (Grimaldi, Savona, Italy harbor))
- Departure from harbour 1
- Arrival at harbour 2
- Departure from harbour 2
- Delivery at dealer (**Note**: (Autocare, US, OR Final Dealer))

During the distribution of the finished vehicle, i-FAST operators can operate on the vehicle (and so, on the battery) until this reaches the harbour, where the only operators authorized to operate on the vehicle are the Grimaldi ones. Finally, the vehicle will reach its final destination to the final dealer (here the Autocare's operators or final dealers will be the only ones who can operate on the vehicle). The operations authorized in the various stages of vehicle distribution are in case of abnormal behavior of the vehicle or battery pack (e.g. temperature threshold). As soon as the vehicle shipment ends, the Controller will stop monitoring the battery.

The following BPMN diagram, superimposable to the main one (figure 5.1.1), can be helpful to understand how operators relates to the battery, and how this can be recorded in the blockchain



Controller
Log authorized attempt to operate on battery
Log unauthorized attempt to operate on battery
Log and send response to operator
Retrieve battery status and operator information
Blockchain
Yes
No
Operator asked permission to operate on battery
Start monitoring
End monitoring
Update Battery/Vehicle Event
Battery / Vehicle Event
FCA
Stellantis DB
Operator wants to operate on battery
Notify Controller
Controller response to operator
Notify Operator
Type?
Yes
No
Event is related to battery operation
Store data/log in DB
Stellantis DB
Store data/log in Blockchain
Blockchain
Operator's smartphone
Operation needed on battery
Authenticate on FCA's system
Ask to operate on battery
Wait for response

Figure 5.2: Whenever an operator wants to correctly operate on a battery, a request, from his/her smartphone, is forwarded to the Stellantis system, which will then forward the request to the controller. The controller, after checking the current battery status on the blockchain, authorizes (or not) the operation creating a log, which will be saved by the Stellantis DB on the blockchain and forworded to the the operator.

5.3 Data processing and GDPR compliance

The battery controller is going to process some personal data of the operator. This action, as suggested by [3], can be expressed by some typical operations: access, store, profiling, and transfer. Giving a look the the above BPMN, the controller in order to authorize the operator, needs to access, profile and store the operator information. According to [3], given this four operations, we need to focus on sevaral GDPR rules and try to answer to this questions:

- **Access**:
  - **Q1**: Does your device deal with sensitive data?
  - **Q2**: Does your device support encryption for operator personal data?
- **Store**:

  - **Q1**: Does your device enable users to delete their data in the original used device?
  - **Q2**: How long will the data be stored?
  - **Q3**: How long it is necessary for processing data through your device?

- **Profiling**:
  - **Q1**: Does your device avoid profiling on the data of the user who is under 18?
  - **Q2**: Does your device deal with profiling of sensitive data?

- **Transfer**:
  - **Q1**: Does your device give the choice of EU-based migration of data?
  - **Q2**: Has your underlying connected device been certifing for their Binding Corporate Rules (BCR) claused by a EU DPA?

These questions are useful in order to give a correct and complete answer to the requests of the GDPR regulation, with a particular look to these articles:

- Art. 31 states that "[...] The processing of personal data by those public authorities should comply with the applicable data-protection rules according to the purposes of the processing.";
- Art. 17 states that "[...] The data subject shall have the right to obtain from the controller the erasure of personal data concerning him or her without undue delay, and the controller shall have the obligation to erase personal data [...]" when this data "are no longer necessary in relation to the purposes for which they were collected or otherwise processed. [...]";
- Art. 22 states that profiling on sensitive data or on users who are under 18 is risky and an encryption mechanism is required (source: [3]);
- Art. 44-47 states that GDPR restrict actors to transfer personal data only inside Europe or the countries holding Binding Corporate Rules (BCR) certifications. The BCR is internal rules (e.g. code of conduct) which are adopted by a community of multinational companies that want to move personal data internationally across various jurisdictions (source: [3]).

5.4 GDPR compliance verified through the blockchain itself

In our model we have the operator, which is the actual data subject (user), the controller (the actor) and a verifier (an authorized third party connected to the blockchain that will take part to the verification of the blockchain, reporting actors not complying with GDPR regulation). All of the three should have an unique ID that refers to a blockchain wallet ID to be registered in the blockchain network ([3] suggests an Ethereum account, but any other account of a known blockchain would be a valid one) or more in general an unique ID obtained after being registered in the Stellantis system, providing a valid identification document (in the case of the user). Once registered, the blockchain's smart contracts may be accessed in order to comply with the GDPR.

As a further note, being the blockchain at the early stages of development, as reiterated in the previous sections, at the moment the only node able to directly add blocks is the Stellantis' central one.
For this reason it is necessary that all the operators, both the internal ones and those belonging to the logistic companies, make reference to this central node. In the future it is certainly desirable that the other nodes, each belonging to one of the logistics companies partner of the process of delivery of the electric vehicle, can directly add blocks to the blockchain, without the need for these to pass through the central node. At that point one can think that every operator of the various sectors trusts only their reference node, sending the latter his requests to access the battery.

5.4.1 GDPR-Operation Contracts

Four contracts, based on the four operations that each actor can perform on user's data. This contracts, proposed by [3], assumes that the GDPR compliance is a boolean field, returned as a result by the smart contract itself along with the actor address and the processed personal data.

Access' smart contract

States that the GDPR compliance is not reached if the actor (the controller in our study case) does not support the encryption of the personal data to be processed. As for the actual encryption algorithms, details are provided in the following chapters.

Access(): Let

- add_a: actor address.
- D_r: set of personal data that must be processed by the actor.
- $encrypt$: boolean value that shows whether the service of actor supports the encryption of personal data or not. For instance, if encrypt is "true", it means that the service offered by actor provides an authentication technique such as a secure login system for protecting sensitive data.

Algorithm 1 The function of access operation

```
    Input: $add_a$, $D_r$, $encrypt$
    Output: $add_a$, $D_r$, $compliance$
1: function ACCESS
2:     $compliance$ = true;
3:     if $encrypt$ == false then
4:         $compliance$ = false;
5:     return($add_a$, $D_r$, $compliance$);
```

Figure 5.3: Algorithm 1 presents the function. If the value of encrypt is "false", the execution of operation violates the GDPR rule legislated in Art. 32(1)(a).

Store's smart contract

Storage operation meet the GDPR requirements if and only if users are able to erase their personal data at any time, and this data will be kept in the blockchain only for the needed time to process them.

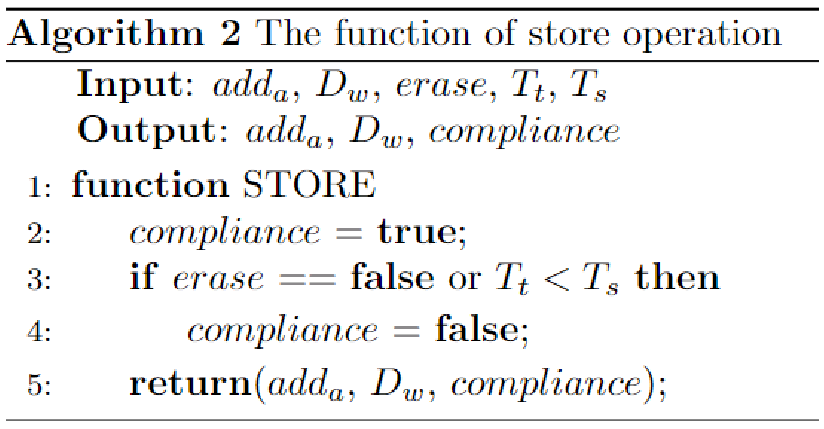
Store(): Let

- add_a: actor address.
- D_w: required personal data.
- T_t: the time totally taken for data processing.
- T_s: the period of time during which personal data will be kept in the storage of actor. After what was said in the chapter 2.2.5, for this storage limitation rule there can be proposed two main solutions:

1. Clearly notify the operator that their data will remain in the blockchain for future checks (or for future data processing) unless, in the future, it will be received an explicit request to delete their personal data stored in the blockchain. In this case there would not be an actual time storage limitation T_s;
2. Delete the personal data from the blockchain, as soon as the battery (vehicle) shipment ends.

I personally preferred to implement solution number one, but with a small change it would be possible, in future, to implement the solution number two, simply by keeping track of blocks that contain personal data of operators, cancelling them once the battery shipment is finished.

- $erase$: a boolean value that clarifies whether the service of actor enables users to erase their data at any time or not. For example, if the value is "true", it means that the service or device has an option for users to delete their data from the local storage of actor.
  As we will see in the next chapters, for the operators it will be possible to ask for data erasure, thanks to the use of a particular type of hash functions, called chameleon hashes.



```
Algorithm 2 The function of store operation
    Input: $add_a$, $D_w$, $erase$, $T_t$, $T_s$
    Output: $add_a$, $D_w$, $compliance$
1: function STORE
2:     $compliance$ = true;
3:     if $erase$ == false or $T_t < T_s$ then
4:         $compliance$ = false;
5:     return($add_a$, $D_w$, $compliance$);
```

Figure 5.4: As represented in Algorithm 2, the execution of store operation complies with GDPR, if erase is "true" and $T_s <= T_t$.

Profiling's smart contract

Profiling operations comply with GDPR requirements if actor supports encryption of user's personal data, and if this one is an adult.

Profiling(): Let

- add_a: actor address.
- D_p: personal data that must be processed.
- $encrypt$: boolean value representing whether the encryption of sensitive data is supported or not.
- $isadult$: boolean value so that if the service of actor performs only profiling operation on user over 18, isadult is "true".

```
Algorithm 3 The function of profiling operation
    Input: $add_a$, $D_p$, $isadult$, $encrypt$
    Output: $add_a$, $D_p$, $compliance$
1: function PROFILING
2:     $compliance = $ true;
3:     if $isadult ==$ false or $encrypt ==$ false then
4:         $compliance = $ false;
5:     return($add_a$, $D_p$, $compliance$);
```

Figure 5.5: Given Algorithm 3, the execution of operation violates GDPR rule (Art. 22) if encrypt is "false" or isadult is "false".

Transfer's smart contract

Transfer's smart contract states that if personal data are willing to be transfered outside of Europe, then compliance with GDPR is reached if and only if this state support BCR's certifications.

Transfer(): Let

- add_a: actor address.
- D_t: personal data that must be transferred.
- loc: country name of data receiver.
- BCR: set containing the list of countries holding BCR certification and EU is a set involving the names of European countries.

Algorithm 4 The function of transfer operation

```
    Input: $add_a$, $D_t$, $loc$
    Output: $add_a$, $D_t$, $compliance$
1: function TRANSFER
2:     $compliance = \textbf{true}$;
3:     if $loc \notin EU$ then
4:         if $loc \notin BCR$ then
5:             $compliance = \textbf{false}$;
6:     return($add_a$, $D_t$, $compliance$);
```

Figure 5.6: As seen from Algorithm 4, if personal data is sent outside Europe and the country of data receiver has not been certified by BCR, the value of compliance is “false”.

According to the European Union, the Binding Corporate Rules (BCR) are data protection standards followed by EU-based corporations when transferring personal data outside the EU within a group of companies or organizations. To establish effective safeguards for data transfers, such policies must encompass all general data protection principles and enforceable rights.

5.4.2 User consent contract

This smart contract allows data subjects to vote and give a final consent in the form of a "yes" or "no" to the execution of operations that have already been claimed through the GDPR-operation contract.
The diagram below (took from [3]), depicts a business process in which an actor registers a claim in the blockchain and a data subject votes to allow the actor’s operations on his personal data. In our BPMN, 5.2.1, this diagram can be attached to the part where the operator (our user) "Asks to operate on battery".
In particular, it is appropriate to attach this diagram between "Notify Controller" (operation in Stellantis DB) and the actual arrival of the data to the controller, in such a way that at that point the user is asked for confirmation from the the Stellantis system just before the data are sent to the Controller, making it clear which information the Controller will be able to access in case of positive vote, **and that this information are going to be encrypted and stored on the blockchain**.

Here there are the operations that can take place on the operator’s personal data. Access, given that the controller will have access to the needed personal data of the operator, store and profiling, since the controller will register, on the blockchain, the attempt of the operator to access the battery. And finally the transfer of personal data to the blockchain itself, in order to record the attempt to access the battery.

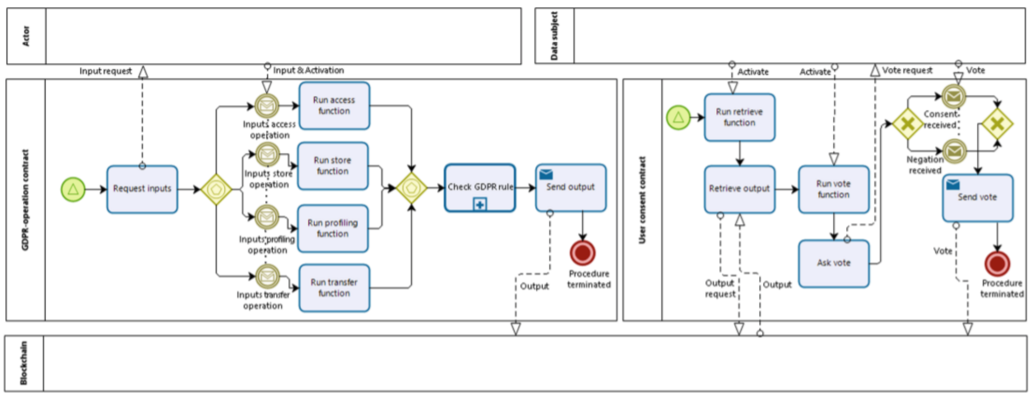


Figure 5.7: Business process of activities in GDPR-operation and user consent contracts.

Sequence diagram

At this point, below it is left a sequence diagram, that can be useful in order to better understand which are the steps to be compiuted by the three actors (Operator, Stellantis' DB and Blockchain), when an operator asks to operate. Even though cryptographic details are discussed in the next sections, at this point can be noted that when encrypted transactions are going to be stored in the blockchain, I decided to leave as the first field of the transaction the public key of the receiver (which will be the only actor able to decrypt the message contained). This choice is useful because every time a new transaction is registered in the blockchain, a Controller in order to know if this is related to the battery it is monitoring, will just have to check if the first field contains its public key.

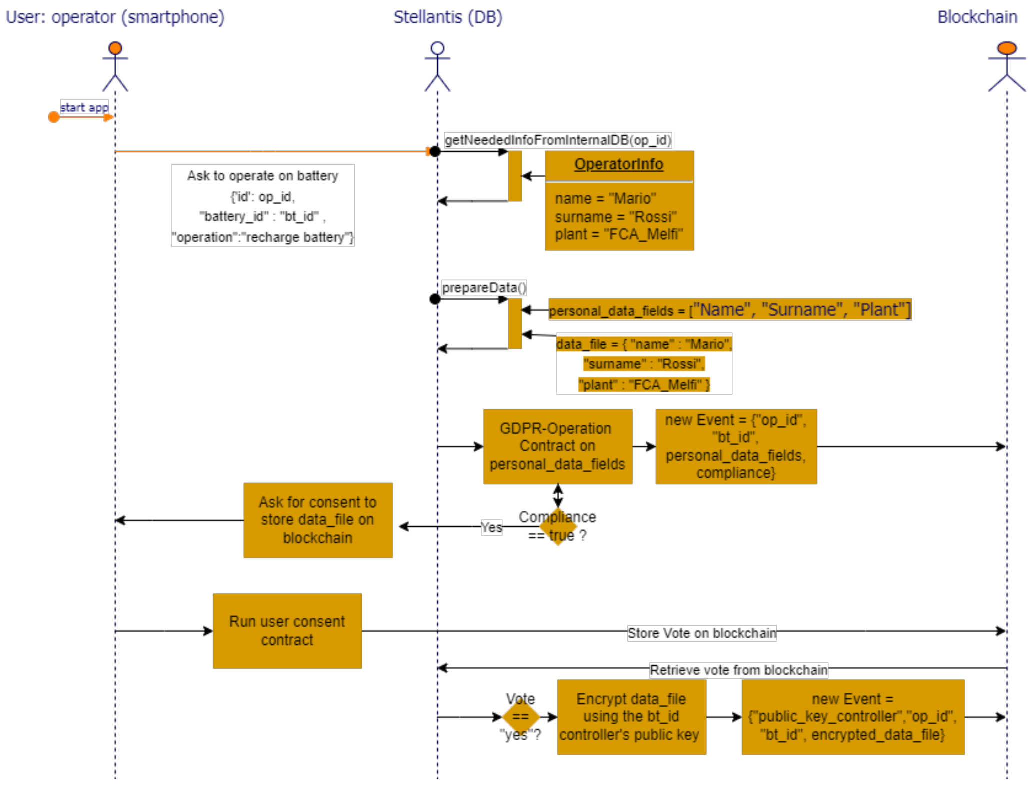

User: operator (smartphone)
Stellantis (DB)
Blockchain
start app
getNeededInfoFromInternalDB(op_id)
Ask to operate on battery {'id': op_id, "battery_id" : "bt_id" , "operation":"recharge battery"}
OperatorInfo
name = "Mario" surname = "Rossi" plant = "FCA_Melfi"
prepareData()
personal_data_fields = ["Name", "Surname", "Plant"]
data_file = { "name" : "Mario", "surname" : "Rossi", "plant" : "FCA_Melfi" }
GDPR-Operation Contract on personal_data_fields
new Event = {"op_id", "bt_id", personal_data_fields, compliance}
Ask for consent to store data_file on blockchain
Yes
Compliance == true ?
Run user consent contract
Store Vote on blockchain
Retrieve vote from blockchain
Vote == "yes"?
Encrypt data_file using the bt_id controller's public key
new Event = {"public_key_controller","op_id", "bt_id", encrypted_data_file}

Figure 5.8: Operator ask to operate: sequence diagram

Note how the fields of the personal data (called personal_data_field) are saved in clear (plaintext) in the blockchain, while the actual values are encrypted using the public key of the controller that manages the battery. This is because, the fields of personal data will be used for future possible verifications by an authorized verifier, as we will see in the next steps.

5.4.3 Submission Contract

The figure below (again, by [3]) depicts a design pattern for interactions between the actor and a smart contract submission. In the blockchain, the actor uses a submission contract to save its address, executed operation, and personal data processed by the operation. To send such information to the blockchain, the contract employs a function named log.
In our battery-assembler system, this sub process can be added to the "Log and send responses to operator", in the Controller (5.2.1).

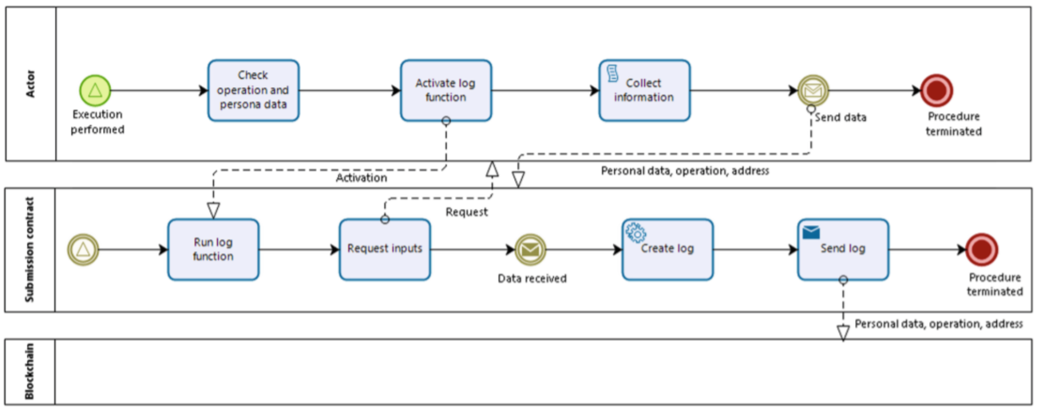


Figure 5.9: Business process of activities performed through submission contract.

5.4.4 Verification contract

The verification of actors is conducted using the business process depicted in the figure below. The list of actors, their operations, and the processed personal data are retrieved from the blockchain once the verification contract's verify function has been enabled. If the actor's performed operation did not get user consent, or if any personal data handled by the operation differs from those already claimed by the actor via the GDPR-operation contract, a violation is clarified by the verification contract. After finding a violation, the verifier casts a vote, which is saved in the blockchain for future reference.
Notably, according to Art. 33 of GDPR, the sub process handling delay is deemed to notify the verifier of any violation or breach within 72 hours. However, under our blockchain-based paradigm, the responsibility of notifying infractions is outsourced to the verifier rather than the actors. The pattern illustrated in the image below can be independently implemented through verifier after completing the procedures described in the business processes.

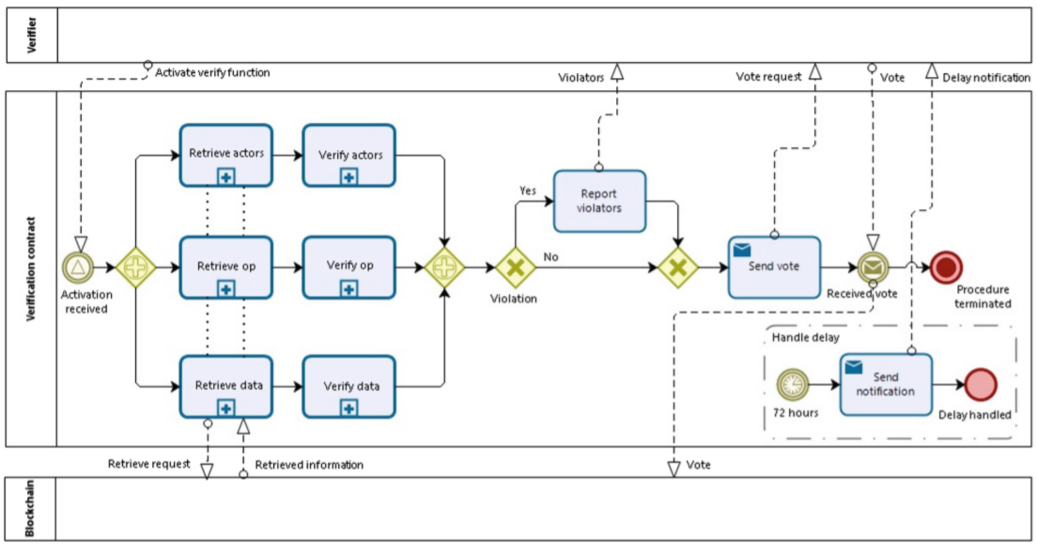


Figure 5.10: Business process of interactions performed through verification contract.

5.5 Cryptography

The previous chapter dealt with managing compliance with the GDPR exploiting smart contracts, taking for granted the aspects of the effective management of data security and cryptographic algorithms, thinking of these as a boolean field called "encrypt".
In this chapter these aspects will be explored, taking as a reference point the algorithms proposed by [6], as they propose a novel model of blockchain to make it suitable for IoT devices like the battery controller. The algorithms exploited will be the following:

- **Lightweighted Ring Signature**: for anonymity and authenticity of the user preserving his privacy while ensuring data integrity. Lightweighted since there will not be used heavy operations such as pairing and exponentiation, to make this more suitable for the controller.

- **Double Encryption**: *ARX ciphers* for symmetric encryption and *public encryption schemes* for asymmetric encryption. ARX ciphers only uses simple arithmetic operations, namely modular addition, bitwise rotation and XOR. First encrypt the data (plaintext) using symmetric key encryption (ARX cipher) and then encrypt the symmetric key itself using a public key (asymmetric).

- **Diffie–Hellman key exchange**: is used to securely exchange cryptographic keys over a public channel.

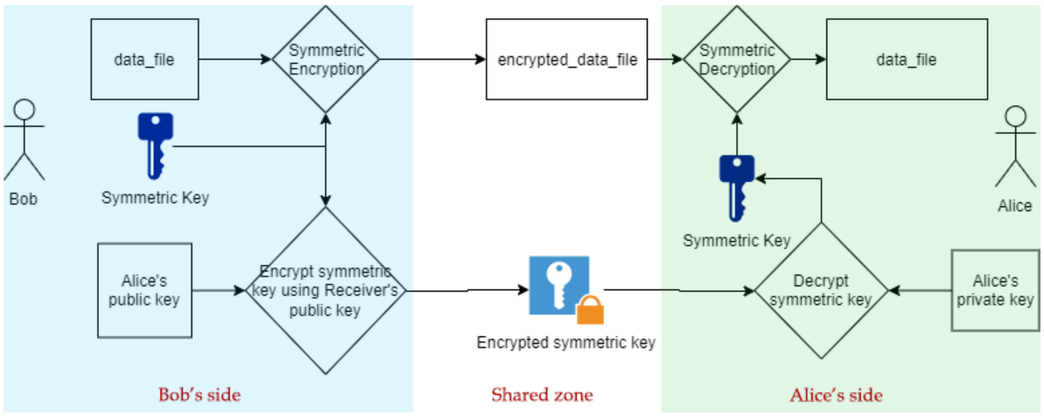


Figure 5.11: Diagram showing double encryption

5.5.1 Cryptographic techniques

Symmetric and Asymmetric encryption

Both symmetric and asymmetric encryption algorithms is used in the system, for different purposes.

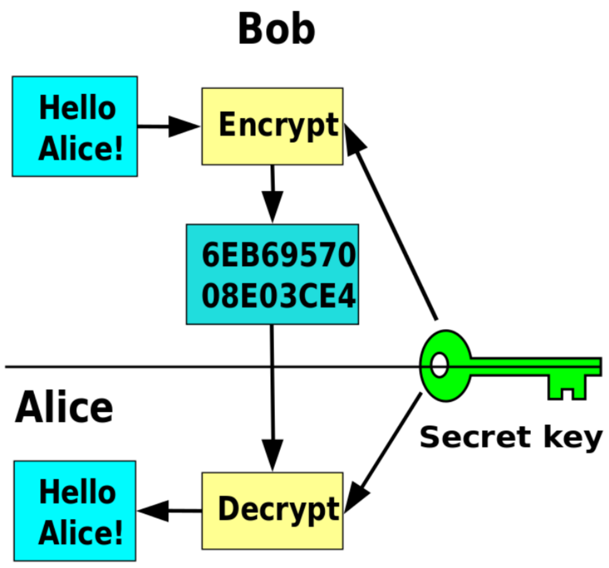


Figure 5.12: Symmetric Encryption Algorithm - Wikipedia

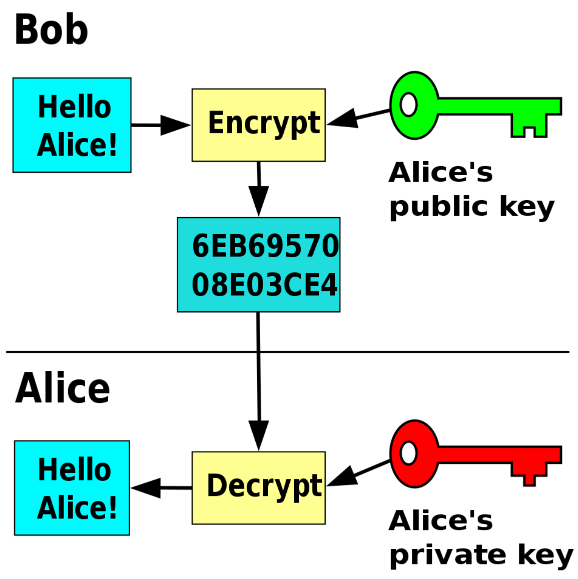


Figure 5.13: Asymmetric Encryption Algorithm - Wikipedia

ARX Encryption

Among all of the ARX Encryption Algorithm, it is suggested the use of SPECK by NSA (National Security Agency). It follows an algorithm description by [29]. Speck can handle a wide range of block and key sizes. A block always consists of two words, which can be 16, 24, 32, 48, or 64 bits long. The appropriate key consists of two, three, or four words. The round function involves two rotations (circular shifts): adding (modulo 2^n) the right word to the left word, xoring the key into the left word, and finally xoring the left word into the right word. The number of rounds is determined by the key size, up to 34 round for a 256 bit key, according to wikipedia.
For major security, in our system is suggested the use of Speck128/256 since the NSA has allowed Speck128/256 for use in U.S. National Security Systems, leading to a big enough block size of 128 bits.

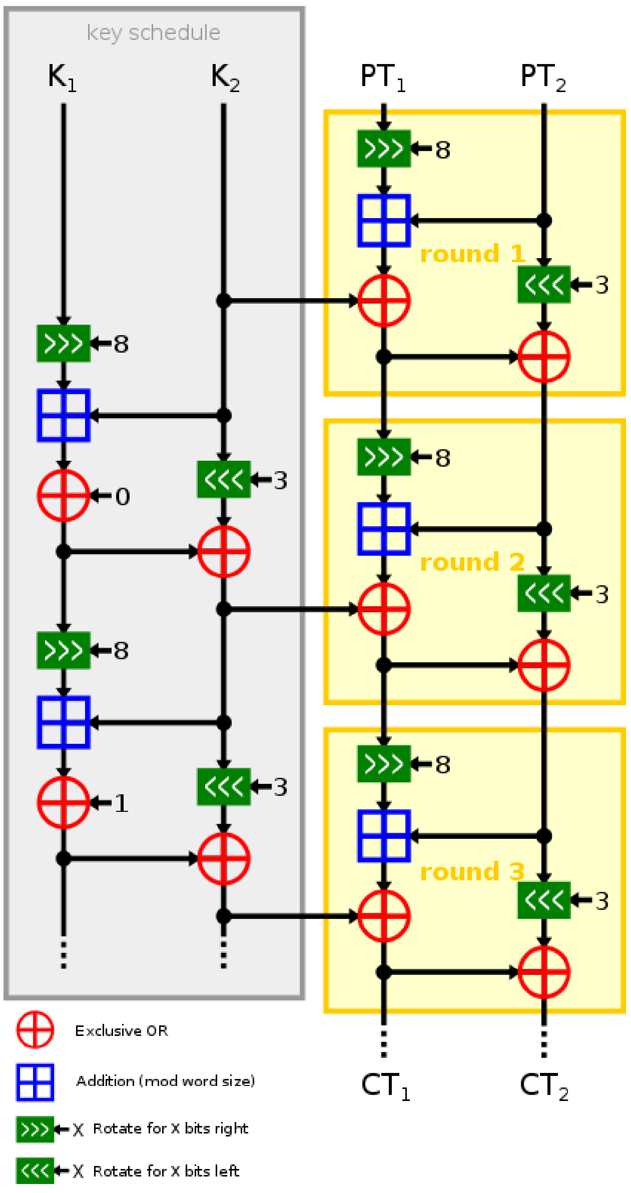


Figure 5.14: Speck algorithm's round - Wikipedia

Lightweighted Digital signature

For authentication purposes, a digital signature is attached to the data. A public-private key pair is assigned to each user. The key pairs used for signing/verifying and those used for encryption/decryption are usually different, in order to achieve more security. In general, each time a message will need to travel from a sender to a receiver, a digital signature will be added. From now on, we will refer to the sender private-public key pair as $\{sk_{s_{priv}}, sk_{s_{pub}}\}$, and to the receiver private-public key pair as $\{rk_{s_{priv}}, rk_{s_{pub}}\}$.

Lightweight Ring signature

The "real" signature is mixed with other groups and no one, but the actual signer, knows which member signed the message. By using this, signers anonymity and signature correctness (where with this last one we mean that a valid signature is always accepted, while a non-valid one is always rejected) is achieved.
In our system, we can suppose that the blockchain will store the public key of all actors willing to partecipate to the ring signature.
Each time an actor want's to apply ring signature to a piece of data, it will perform a request to the blockchain. This request will contain his/her public key (the sender public key $sk_{s_{pub}}$). After receiving the request, the blokchain will send back a certain amount of other public keys of actors that decided to partecipate to other ring signature in the past, eventually it will store the $sk_{s_{pub}}$ (the key of the requester) in the list of available public key willing to partecipate to ring signature (if it was not already available in the list).
For example, the number of public keys the blockchain could give as a respostense for the ring signature could be seven, as in Monero, for a total of eight public keys if we include the sender public key too. A response by the blockchain could be: $\{sk_{s_{pub}}, sk_{1_{pub}}, sk_{2_{pub}}, sk_{3_{pub}}, sk_{4_{pub}}, sk_{5_{pub}}, sk_{6_{pub}}, sk_{7_{pub}}\}$.

Diffie-Hellman Key Exchange

Diffie-Hellman Key Exchange is exploited in order to secretly exchange the public keys needed in the above algorithms.

5.6 Four algorithms to implement data security

In general, data communication between sender and receiver takes place with four algorithms. The first two performed by the sender, while the last two will be performed by the receiver.

Sender's algorithm

1. **Data Encryption**: the "plaintext" data, e.g *data_file* is encrypted obtaining an *encrypted_data_file*.
2. **Signature**: a signature is added to the *encrypted_data_file*.

Encrypted data alongside with signature is then sent to the receiver.

Receiver's algorithm

1. **Data Decryption**: *encrypted_data_file* is decrypted obtaining back *data_file*.
2. **Signature verification**: the signature is verified, making sure the data has been sent by the sender, and has not been tampered with.

5.6.1 Algorithms' specifications

For this section, algorithms by [6] were taken as a reference.

Algorithm 1: Data Encryption

Symmetric key k_{sym} is used to encrypt the $data\_file$, obtaining a cyphertext (C). After this, the k_{sym} itself is encrypted (double encryption technique) by using the asymmetric public key of the receiver rk_{pub} obtaining an encrypted symmetric key (C_k). At the end the cyphertext C along with the encrypted symmetric key C_k is sent to the receiver.

Algorithm 1 Data Encryption

```
1: procedure User wants to encrypt the data
2:     Generate symmetric key $k_{sim}$
3:     encrypted_data_file < Encrypt the data_file using the symmetric key
4:     $C_{Ksym}$ < Encrypt the symmetryc key using the public key of the receiver
5: end procedure
```

Algorithm 2: Ring Signature and Public Key Sharing

Algorithm used to add a digital signature to the data that are exchanged, so as to guarantee authentication of the sender and integrity and non-repudiation of the message.
If anonymity is required, then the sender's signature is mixed with other users' signatures (from other accounts who also want to add ring signature to their transactions) and then sent it over the network. No one will be able to identify the original signer of the ring group. The set of users who also wish to apply ring signature will be provided by the network itself.

Algorithm 2 Signature and Public Key Sharing

```
1: procedure USER WANTS TO SIGN THE DATA AND SHARE ITS PUBLIC KEY
2:     Generate asymmetric public-private key pair $\{sk_{s_{priv}}, sk_{s_{pub}}\}$
3:     $hash_p <$ calculate hash of encrypted_data_file
4:     Digitally sign $hash_p$ using the private key
5:     Share the public key using Diffie-Hellman key exchange
6:     if $AnonimityRequired = True$ then
7:         Add ring signature mixing the signature with other signer from the blockchain
8:     end if
9: end procedure
```

Algorithm 3: Data Decryption

Firstly the receiver will decrypt the encrypted symmetric key C_k, using his own private key rk_{priv}, obtaining the same symmetric key k_{sym} used by the sender. After that, the ciphertext C will be decrypted symmetrically by using the symmetric key just obtained, deriving the original $data_file$ in plaintext.

Algorithm 3 Data Decryption

```
1: procedure USER WANTS TO DECRYPT THE ENCRYPTED DATA
2:     $k_{sym} <$ decrypt $C_{Ksym}$ using your private key. ▷ Note: you can decrypt it only if you're the correct receiver
3:     data_file < decrypt encrypted_data_file using $k_{sym}$
4: end procedure
```

Algorithm 4: Signature Verification

Last algorithm in the list, the receiver will verify if the signature is valid or not.

Algorithm 4 Signature Verification

```
1: procedure USER WANTS TO VERIFY THE SIGNATURE
2:     $hash_c <$ calculate hash of the encrypted_data_file
3:     Extract $hash_p$ using the public key of the sender
4:     if $hash_c = hash_c$ then
5:         return encrypted_data_file                    ▷ The signature is valid
6:     else
7:         return "Signature incorrect"
8:     end if
9: end procedure
```

5.7 Implementation

This section, exploiting the four cryptographic algorithms mentioned above, shows how to protect the transmission of data to and from the controller and of the data entering the blockchain, with particular attention to when this data are personal ones. In the latter case, these personal data will be encrypted before being saved in the blockchain and only the controller associated with the battery will be able to decrypt them.

5.7.1 Operator asks to operate on a battery

Referring to BPMN in figure 5.2, the operations between the smartphone and the Stellantis systems are considered safe and encrypted. In addition a signature by the smartphone would be suggested in order to assure Authentication of data (Anonymity would also be achieved if this algorithm is used in ring mode). Below it is provided a description of a possible solution that guarantees the security of these data also in the Controller and in the blockchain (5.2.1).

Channel to be secured

In this case the channel to be secured is the one connecting the Stellantis DB to the Controller, through the "Notify Controller" activity.

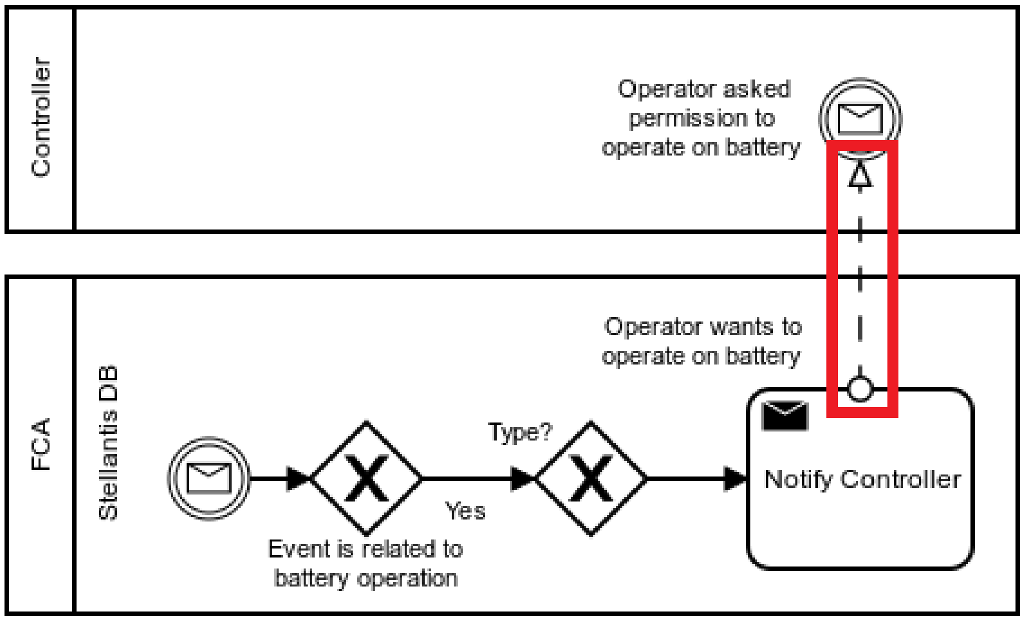

Controller
Operator asked permission to operate on battery
FCA
Stellantis DB
Operator wants to operate on battery
Type?
Yes
Event is related to battery operation
Notify Controller

Figure 5.15: Zoom of the area of interest in the general BPMN. In red, the channel to be secured

The algorithms to be applied are all of the four specified above (5.6), where **sender**'s role is interpreted by the Stellantis DB, while the **receiver**'s role is played by the Controller. In this case, in the algorithm 2, no ring signature is needed since the data are directly coming from the Stellantis DB.
In this phase, the Stellantis DB will notify the Controller that an operator wants to perform a given operation on the battery, e.g. "Charge the battery". It will provide to the Controller this information stored in a new transaction in the blockchain:

- **Unique ID of the operator**;
- **Strictly needed operator personal information** (the request will comply with GDPR thanks to the smart contracts in the precedent chapters and since the Operator will be informed that this personal information are going to be collected and processed by the controller, before they are actually sent to the controller);
- **Type of operation** the operator wants to perform on battery (needed just necessary for subsequent logs).

Note that in this phase, the Controller will just receive the list of "Strictly needed operator personal information", not the content itself, which will be collected in the next step. For example, the list of personal information will be of this type:

"Name, Surname, Plant" and will not contain the actual values, such as "'Mario', 'Rossi', 'FCA MELFI'". The actual values, as soon as the user consent is recorder in the blockchain by the proper smart contract, will be encrypted and stored in the blockchain, and only the correct Controller will be able to encrypt them.

Steps in this phase

In the Stellantis' DB side:

1. Create a new log, containing the needed information (e.g., {ControllerID, OperatorID, {List of Operator's personal data field}, typeOfOperation}). The new log is now called data_file;
2. Once the GDPR-Operation Contract confirm compliance, digitally sign the data_file using algorithm 2, and store it along with the signature in a new transtaction in the blockchain;
3. In the next step, the database will be ready to encrypt, digitally sign and save the operator's personal data once the final consent is reached throught the user consent smart contract.

5.7.2 Stellantis DB storing data on blockchain

This final step is important, as it is the segment where the data are saved on the blockchain. In this phase, we have a **sender**, the Stellantis' DB, and a **receiver**, the blockchain.

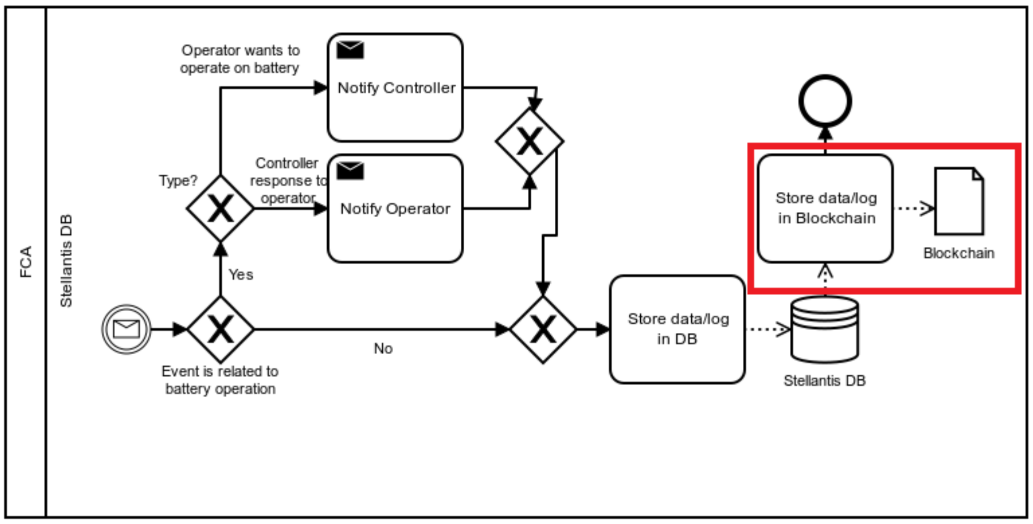

FCA
Stellantis DB
Operator wants to operate on battery
Notify Controller
Type?
Controller response to operator
Notify Operator
Yes
Event is related to battery operation
No
Store data/log in DB
Stellantis DB
Store data/log in Blockchain
Blockchain

Figure 5.16: Zoom of the area of interest in the general BPMN. In red, the channel to be secured

In this case, there is no need to apply the Algorithm 3 (Data Decryption) and Algorithm 4 (Signature Verification), which will be applied later, when the Controller will collect this data in future steps.
The goal here, is to store the new data / log in the blockchain, regarding the given battery.

- If the data contains personal information, encrypting them with Algorithm 1 (the symmetric key used to encrypt the personal data is encrypted with the battery's controller public key and stored along with the new data, so that only the given controller will be able to encrypt them later);
- In any case, sign the data_file (or encrypted_data_file in the case of personal informations) using Algorithm 2 (without the need of ring signature, a simple signature is enough since we suppose there is only one Stellantis DB).
- Once the block is added, any node of the blockchain will be able to access the battery status / data, a part from the personal informations' which will be accessible only by the proper battery Controller.

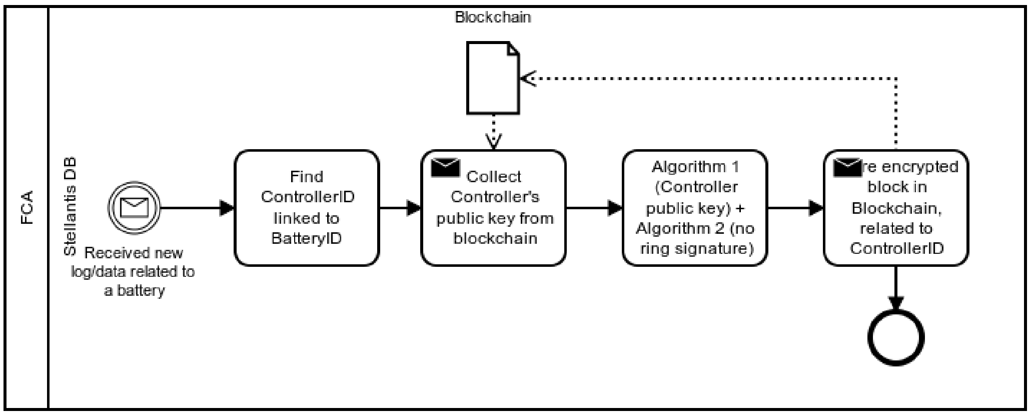

Blockchain
FCA
Stellantis DB
Received new log/data related to a battery
Find ControllerID linked to BatteryID
Collect Controller's public key from blockchain
Algorithm 1 (Controller public key) + Algorithm 2 (no ring signature)
re encrypted block in Blockchain, related to ControllerID

Figure 5.17: Workflow

Note: data/log could also be the needed operator's information of the operator than wants to operate on battery.

5.7.3 Controller collecting battery status and operator information

The goal of this section is to guarantee the security of data when the Controller is collecting battery status and operator information from the blockchain (5.2.1).
Just to give a context to this activity, here the Controller will collect the last Container or Vehicle Event thus becoming aware of the current state and position of the battery. Then, using the unique code of the operator (received in the previous step from the Stellantis DB), it will collect his/her data of interest, such as the plant for which it works, from the transaction of interest in the blockchain (remember: only the proper Controller will be able to decrypt the data stored by the Stellantis DB). At that point he will be able to tell if the operator is authorized to operate on the battery or not, creating a log that will inform the DB stellantis, the blockchain and finally the operator itself.

Channel to be secured

In this case the channel to be secured is the one connecting the Blockchain to the Controller, through the "Retrieve battery status and operator information" activity.

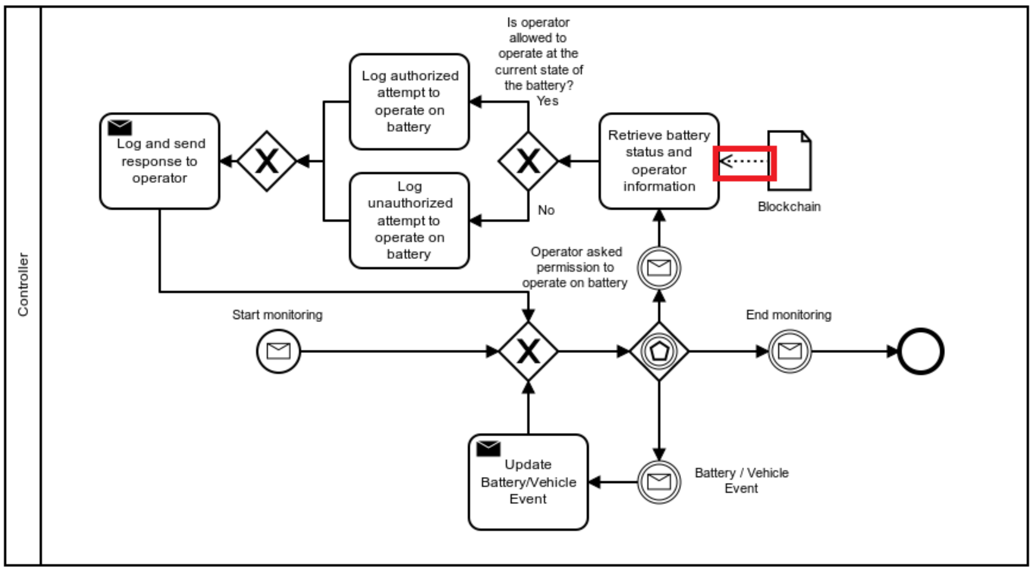

Controller
Is operator allowed to operate at the current state of the battery?
Yes
No
Log authorized attempt to operate on battery
Log unauthorized attempt to operate on battery
Log and send response to operator
Retrieve battery status and operator information
Blockchain
Operator asked permission to operate on battery
Start monitoring
End monitoring
Update Battery/Vehicle Event
Battery / Vehicle Event

Figure 5.18: Zoom of the area of interest in the general BPMN. In red, the channel to be secured

The algorithms to be applied here are just the number 3 and 4, since the first two have been already applied by the Stellantis DB when storing the data on the blockchain itself. Here the **sender**'s role is interpreted by the Blockchain, while the **receiver**'s role is played by the Controller.
The goal of algorithm 3 and 4 applied here are to be sure that the data stored on the blockchain have been stored by the Stellantis DB itself, and decrypt the personal data (only the correct controller can decrypt the symmetric key, and by verifiying the signature the controller is sure that the data are not tampered and were written by the Stellantis DB).

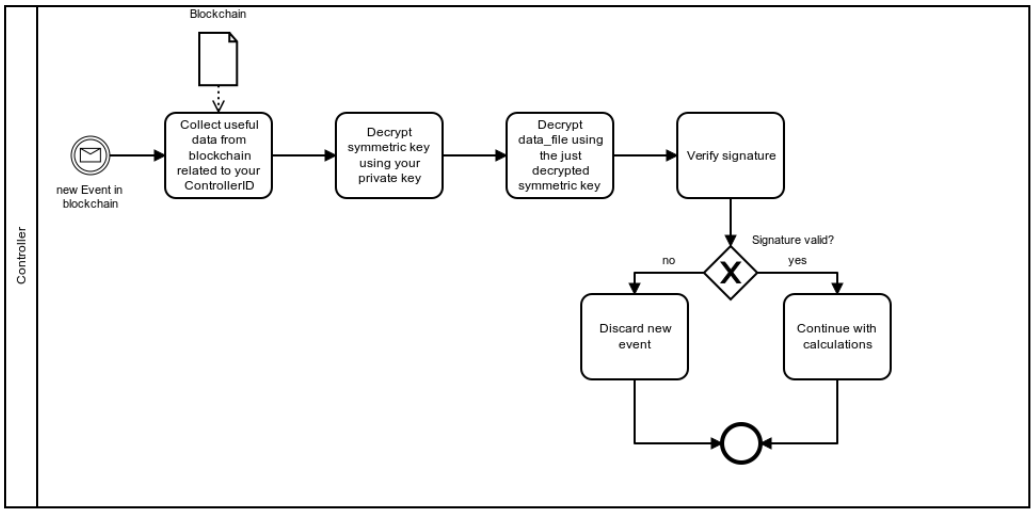

Controller
Blockchain
new Event in blockchain
Collect useful data from blockchain related to your ControllerID
Decrypt symmetric key using your private key
Decrypt data_file using the just decrypted symmetric key
Verify signature
Signature valid?
no
yes
Discard new event
Continue with calculations

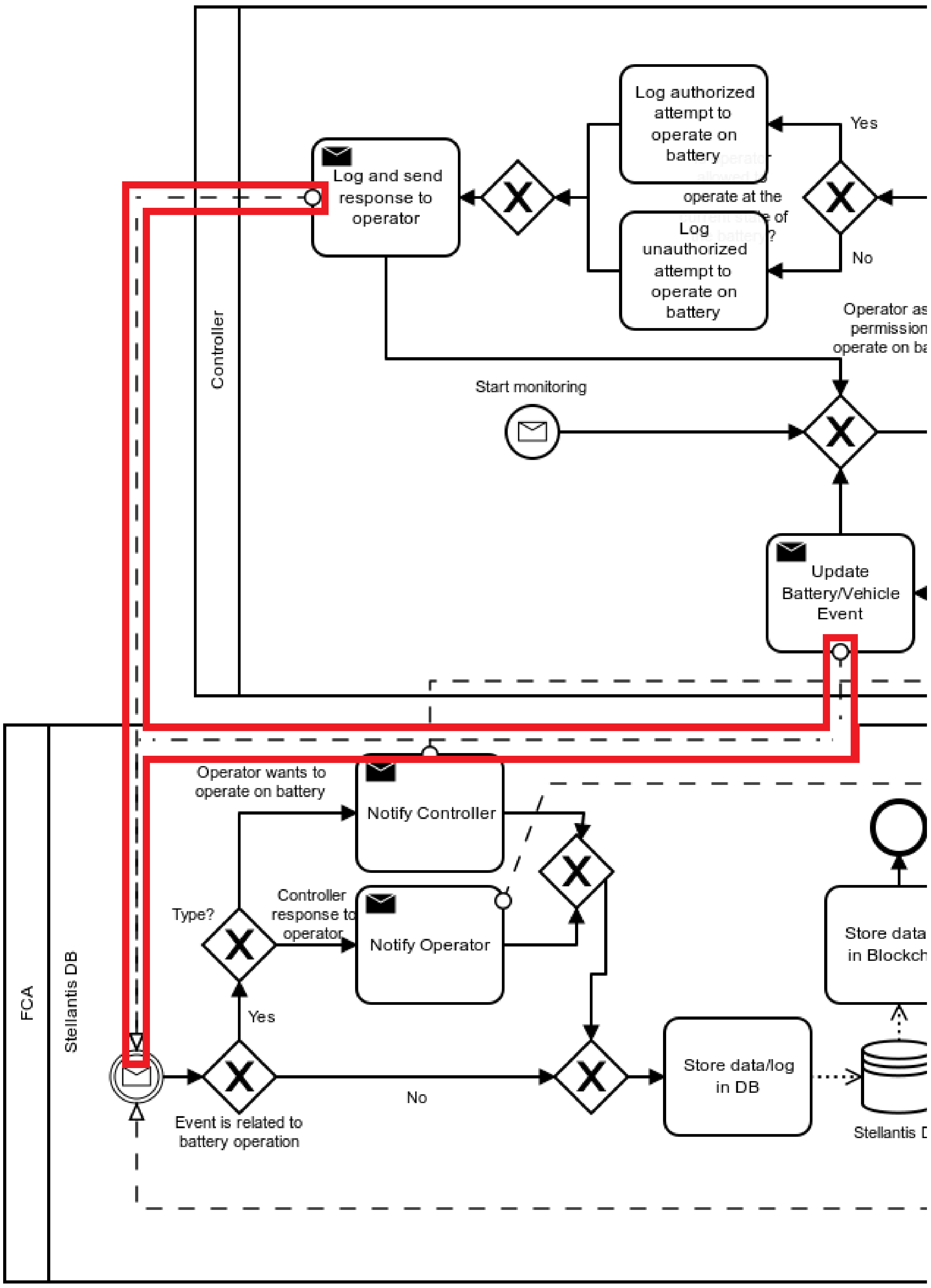
Figure 5.19: Workflow

5.7.4 Comunication from Controller to Stellantis DB

Here we can accumulate two Controller events, from the point of view of security algorithms, as in both there is a communication that goes from the **Controller (Sender)** and the **Stellantis servers (Receiver)**. This events are the

- "Log and send response to operator" which happen after the Controller has elaborated a response to be recorded on the blockchain and to be given to the operator;
- "Update Battery/Vehicle Event" which happens when the battery controller receive a new update event to be stored in the blockchain.

In this case, the ring signature can be applied (in algorithm 2), so that the given battery controller can sign data using its public key, and the one of other 7 battery controllers that already applied for ring signature before. By doing so, we would add an extra level of difficulty for an outside attacker trying to figure out, for example, which battery an employee worked on to use this information to his advantage.



Controller
Log authorized attempt to operate on battery
Yes
Log and send response to operator
operate at the
Log unauthorized attempt to operate on battery
No
Operator as permission operate on ba
Start monitoring
Update Battery/Vehicle Event
FCA
Stellantis DB
Operator wants to operate on battery
Notify Controller
Type?
Controller response to operator
Notify Operator
Store data in Blockch
Yes
Event is related to battery operation
No
Store data/log in DB
Stellantis D

Figure 5.20: Zoom of the area of interest in the general BPMN. In red, the channel to be secured

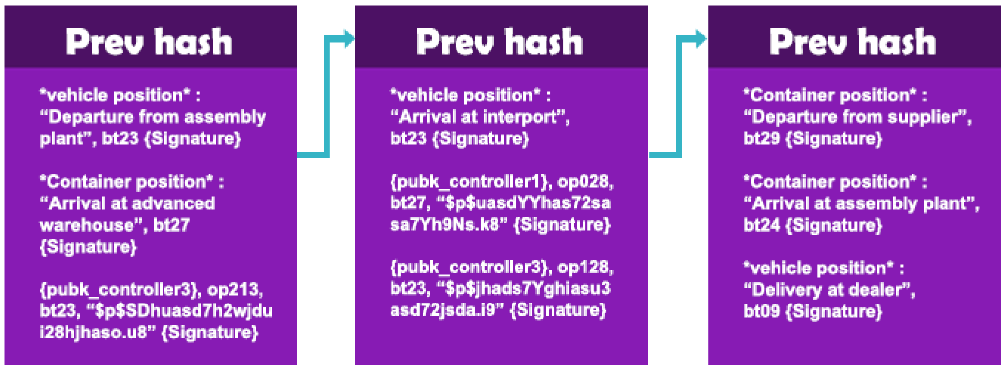

Prev hash
vehicle position : "Departure from assembly plant", bt23 {Signature}
Container position : "Arrival at advanced warehouse", bt27 {Signature}
{pubk_controller3}, op213, bt23, "pSDhuasd7h2wjdu i28hjhaso.u8" {Signature}
Prev hash
vehicle position : "Arrival at interport", bt23 {Signature}
{pubk_controller1}, op028, bt27, "$p$uasdYYhas72sa sa7Yh9Ns.k8" {Signature}
{pubk_controller3}, op128, bt23, "$p$jhads7Yghiasu3 asd72jsda.i9" {Signature}
Prev hash
Container position : "Departure from supplier", bt29 {Signature}
Container position : "Arrival at assembly plant", bt24 {Signature}
vehicle position : "Delivery at dealer", bt09 {Signature}

Figure 5.21: A possible section of the blockchain where battery status/data are visible to anyone in the network, and personal data are encrypted

5.8 Final Security Evaluation

Below are listed some evaluations about the final system that takes advantage of the features listed above to comply with the GDPR regulation. As it could be deduced, the final system exploits smart contracts to deal with GDPR exploiting a "by design" architecture, while applying cryptographic algorithms on data (leading to a "non by design" architecture). This is because we must aim to develop a system that makes a new blockchain collaborate with an existing database (the Stellantis DB) which already have most of the data inside of it.

The CIA's (Central Intelligence Agency) Triad of confidentiality, integrity, and availability is widely regarded as the foundation of information security. One or more of these basic concepts can be applied to any security control and security risk. To be considered comprehensive and complete, a security program must appropriately address the entire CIA Triad.
In our system, we achieve this (and other security requirements) in this way:

- **Confidentiality**: exploiting Asymmetric Encryption;
- **Integrity**: exploiting hashing of data block while signing data;
- **Availability**: only a limited set of transactions are accepted in the blockchain, since each transaction is passing through the Stellantis DB before being recorded in the blokchain itself. There is no way to "dos" the blockchain, since any invalid transaction can be blocked by the Stellantis DB before being recorded;

- **Anonimity**: each Controller can exploit ring signature when storing data in the blockchain;
- **Authentication**: exploiting digital signature;
- **Non-repudiation**: exploiting digital signature.

5.9 Regulation compliance's checklist

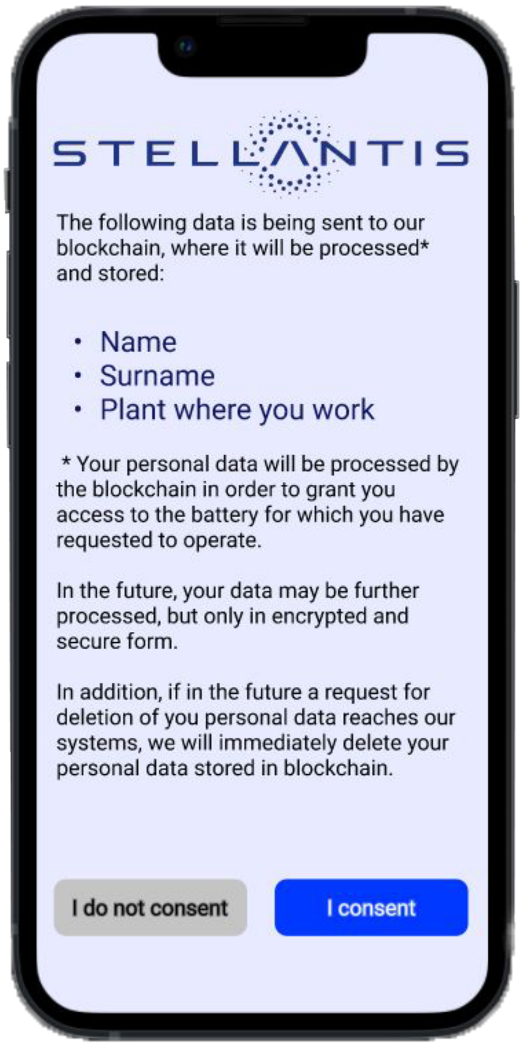
The following is a list of the main requirements by the GDPR, which we have discussed in previous chapters, and how these are met in the proposed system.
Although many of these requests are met thanks to what has been done in the previous sections, in order to satisfy the other requests it is sufficient to expose to the user, with transparency, what are the operations that the system does on their personal data.
This section will end by showing an example of a screen to be shown to the user, at the final stage of requesting consent on the transfer of personal data to the blockchain.

- **Right to be informed** (transparency): compliance with Article 13 & 14 of GDPR is achieved by informing the users (operators in our study case), about which data are going to be stored in the blockchain, providing them with a detailed, concise and clear list of their personal information that will be transmitted to the blockchain in the case that they provide permission through the User consent contract.
- **Purpose limitation**: compliance with Article 5(1)(**b**), of the GDPR, closely connected with the above law, is obtained thanks to the fact that when asking for user consent to process and store their data in the blockchain, together with the list of personal data, it will be clearly indicated why their data will be processed, communicating to the data subject the related implications of their data to be stored in a blockchain, such as that the processing is not limited to the original transaction but that their personal data will continue to be processed thereafter, even if in a way that does not allow anyone to be able to decipher their personal information (thanks to the use of hash functions).
- **Data minimisation**: in our case, there is no processing of special categories of data, since when it comes to personal data, only the name, surname and the sector/plant where the operator works are stored. However, in future work, additional personal information may need to be stored, but the important thing is that this will be **strictly necessary and relevant** information.

- **Data accuracy**: compliance with Article 5(1)(**d**) of the GDPR is achieved thanks to the usage of chameleon hashed (of which I talk about in the next section). Infact, having regard to the purposes for which this personal data are processed, when an update to this data is required, transactions containing old or incorrect information will be invalidated, creating new corrected/updated transactions.
- **Storage limitation**: as already discussed in the store's smart contract 5.4.1 and just before in the purpose limitation requirement, compliance with Article 5(1)(**e**) of the GDPR, is obtained by clearly notify the operator that their data will remain in the blockchain for future checks (or for future data processing) unless, in the future, it will be received an explicit request to delete their personal data stored in the blockchain.
- **Accountability**: Article 5(2) of the GDPR is respected since we are able to demonstrate compliance with article 5, and an external authroized verifier can join the blockchain and check the outcome of the verification smart contract. In our help, in addition, comes the fact that the blockchain we use in particular is a permissioned one, so it is possible to track the identity of the people/companies (even the logistic ones) who are behind a specific node of the blockchain.

Regarding the **right to erasure** or ammend, it is achieved through the usage of chameleon hashes. Given the importance of this law, in the particular case in which it is applied in a blockchain, below is a section with an in-depth analysis.

Below, a draft of what the screen should look like when operators are asked for their consent to store data in blockchain. This vote, asked during the User Consent Contract, will be stored in the blockchain itself.



STELLANTIS

The following data is being sent to our blockchain, where it will be processed* and stored:

- Name
- Surname
- Plant where you work

* Your personal data will be processed by the blockchain in order to grant you access to the battery for which you have requested to operate.

In the future, your data may be further processed, but only in encrypted and secure form.

In addition, if in the future a request for deletion of you personal data reaches our systems, we will immediately delete your personal data stored in blockchain.

Figure 5.22: Proposed draft of a user consent screen

5.9.1 Right to erasure via chameleon hash

As explained in chapter 3.3.3, chamaleon hashes allow the implementation of the right to erasure (needed in order to comply with GDPR) in the blockchain.
A good hash function should be **one-way** (no way to obtain back an input from the output of the hash function), **collision resistant** and lead to a **short output**.
In summary, when a generic hash function is used, it is expected that with a given input, only one output can be obtained: but this is not always the case, since, albeit very difficult and rare to obtain, there can be cases of collision, where two different inputs lead to the same output.

A chameleon hash is a cryptographic hash function that contains a trapdoor: without the trapdoor, it should be really hard to find collisions, but knowledge of the trapdoor information allows collisions to be generated efficiently.
Now, in blockchain when a new block is created, the hash of the precedent block is stored in the header of the new block, and this is the reason why it is impossible to modify a transaction in a block of a blockchain without all the other blocks succeding this one being modified, therefore without the system realizing that something wrong happened.

Someone that is not aware of the trapdoor, has the same difficulties to calculate an hash collision of the transaction as for any other hash function.
This chameleon hash function come along with a collision function, that given as input the current "message" (the transaction in the case of blockchain) and the trapdoor, give as output a new "message2" that lead to the same output when appliying that hash function. A pseudocode will follow:

Hash Function:
Given as input the message to be hashed, the output is the hashed message (H) and the trapdoor (TD) to easily generate collisions.
$\{H, TD\} = ChamaleonHash(Message)$

Collision finder function:
$CollisionMessage = Collision(Message, TD)$

Where $CollisionMessage$ is a new message, such that:
$ChamaleonHash(CollisionMessage) = H = ChamaleonHash(Message)$

Application in our blockchain

Let's suppose an operator appeals to the GDPR's right to be forgotten or right to amend and a Controller is still monitoring (or has monitored in the past) a battery that has been handled by that operator. In that case, in the blockchain there is at least one transaction containing some personal information of the operator (see chapters above).
The solution, is to exploit the chameleon hash as cryptographic hash function in our blockchain. When a new block is created, then the hash of the precedent block, that will be stored in the header of the new block, will be calculated using a chameleon hash function.
Then each time a new transaction containing some personal data is created by the Stellantis systems, along with this new transaction it is stored another transaction containing the trapdoor to edit the first one. Since both transactions will

be encrypted and only decryptable by the proper Controller, whenever an operator appeal to be forgotten will reach the Stellantis system, this system will trigger the proper Controller, asking to substitute the transaction containing the operator's personal information with another one that has the same hash output. This will invalidate the transaction, that will no more contain the personal information, but just a random message leading to the same hash value. If the request was to delete the information, there will be no other steps to be performed, otherwise if the request was for modification of this data, then a new transaction with the updated information will be created. This will correctly comply with the GDPR's right to be forgotten and right to amend, while still having a fully working blockchain. In addition, it is possible to assume that a copy of the generated trapdoor can also be safely saved in the Stellantis database. In this way, if a request for cancellation or modification should ever reach the system when the vehicle has been already delivered to the final dealer and therefore when the Controller has stopped monitoring, the Stellantis central node may be able to invalidate the transaction containing the personal data to be deleted, even though it is not able to decrypt the personal data contained in that transaction.
An illustrative example is provided below.

Let's suppose that a request for data erasures reaches the Stellantis' systems, from a certain operator called Mario Rossi, of the FCA plant in Melfi and that some of his personal data are in the blockchain (for the sake of clarity, in the image this data are not encrypted).

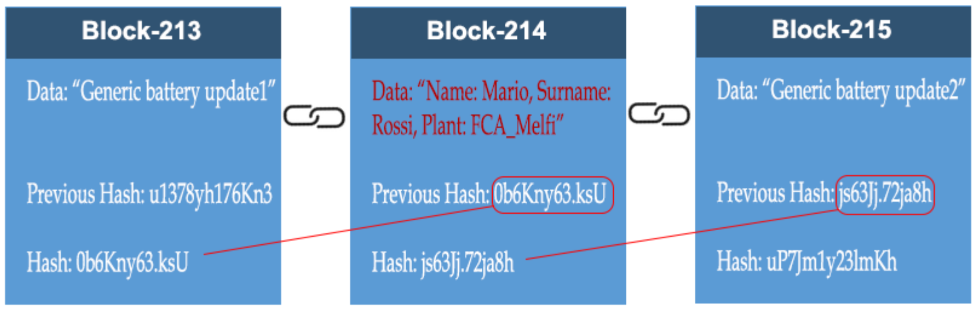

Block-213
Data: "Generic battery update1"
Previous Hash: u1378yh176Kn3
Hash: 0b6Kny63.ksU
Block-214
Data: "Name: Mario, Surname: Rossi, Plant: FCA_Melfi"
Previous Hash: 0b6Kny63.ksU
Hash: js63Jj.72ja8h
Block-215
Data: "Generic battery update2"
Previous Hash: js63Jj.72ja8h
Hash: uP7Jm1y23lmKh

Figure 5.23: Section of blockchain containing personal data to be erased

Thanks to the knowledge of the trap-door associated to the hash value (for example *TD* = *"ka7wn2bIA8.ke"*), a collision message is generated.

CollisionMessage = Collision("Name: Mario, Surname: Rossi, Plant: FCA_Melfi", "ka7wn2bIA8.ke")

The obtained meaningless collision message, e.g. *CollisionMessage = "kasnq7h3bash22123hasd2"*, when hashed is leading to the same hash value of the original message.

ChamaleonHash("kasnq7h3bash22123hasd2") = ChamaleonHash("Name: Mario, Surname: Rossi, Plant: FCA_Melfi", "ka7wn2bIA8.ke") = "js63Jj.72ja8h"

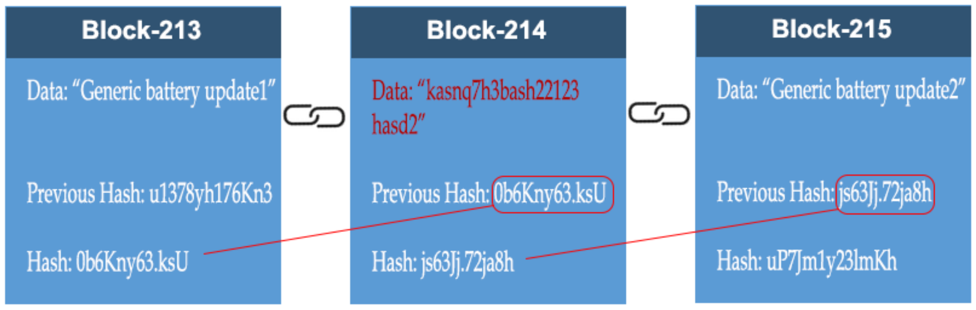

Block-213
Data: "Generic battery update1"
Previous Hash: u1378yh176Kn3
Hash: 0b6Kny63.ksU
Block-214
Data: "kasnq7h3bash22123 hasd2"
Previous Hash: 0b6Kny63.ksU
Hash: js63Jj.72ja8h
Block-215
Data: "Generic battery update2"
Previous Hash: js63Jj.72ja8h
Hash: uP7Jm1y23lmKh

Figure 5.24: Same section of blockchain, after amend of data, with same hash value

5.10 Final Complete BPMN

Although visually cluttered, since there are all the actors participating in the final result, and with the presence of all the smart contracts we have talked about in the previous chapters, I have decided to leave below what the final resulting BPMN is looking like.

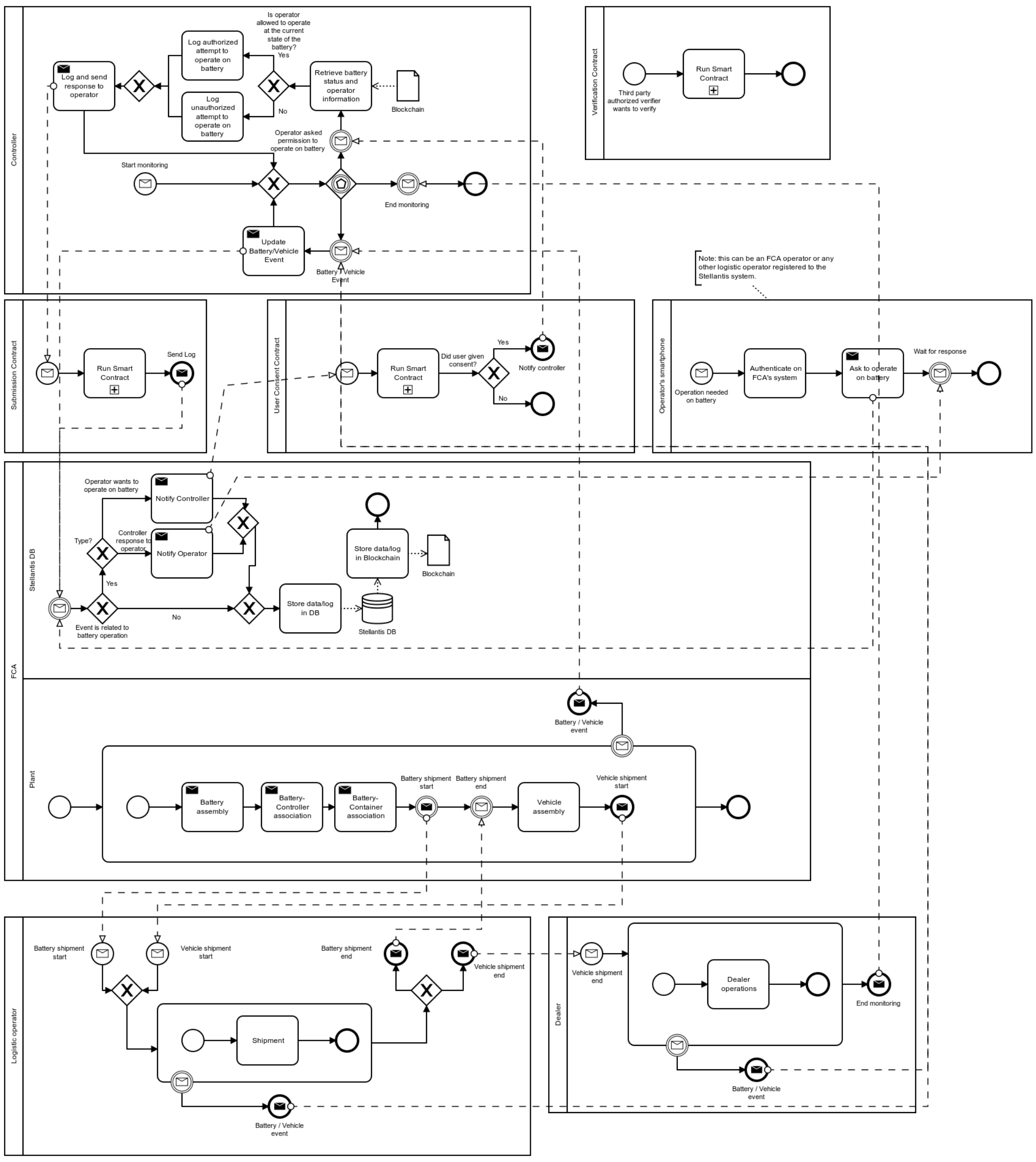

Controller
Is operator allowed to operate at the current state of the battery?
Yes
No
Log authorized attempt to operate on battery
Log unauthorized attempt to operate on battery
Log and send response to operator
Retrieve battery status and operator information
Blockchain
Operator asked permission to operate on battery
Start monitoring
End monitoring
Update Battery/Vehicle Event
Battery / Vehicle Event
Verification Contract
Third party authorized verifier wants to verify
Run Smart Contract
Note: this can be an FCA operator or any other logistic operator registered to the Stellantis system.
Submission Contract
Send Log
User Consent Contract
Did user given consent?
Notify controller
Operator's smartphone
Operation needed on battery
Authenticate on FCA's system
Ask to operate on battery
Wait for response
FCA
Stellantis DB
Operator wants to operate on battery
Notify Controller
Controller response to operator
Notify Operator
Type?
Event is related to battery operation
Store data/log in Blockchain
Store data/log in DB
Stellantis DB
Plant
Battery / Vehicle event
Battery assembly
Battery-Controller association
Battery-Container association
Battery shipment start
Battery shipment end
Vehicle assembly
Vehicle shipment start
Logistic operator
Vehicle shipment start
Battery shipment end
Vehicle shipment end
Shipment
Dealer
Dealer operations
End monitoring

Figure 5.25: Final BPMN

5.11 Expected consumption and performance

At the current state of study, it was not possible to physically access the Stellantis' blockchain, in order to add the proposed implementation.
For this reason, it was not possible to calculate the actual performance and consumption. However, I will limit myself in reporting what are the expected consumption, based on the use of the proposed smart-contracts.

An initial prototype was built in the paper [3] through Ropsten, which is a public blockchain test network, while the proposed smart contracts were implemented on Ethereum using Solidity, an object-oriented programming language designed precisely to create smart contracts. These first tests are useful for us as the blockchain Stellantis is working on is also based on an Ethereum client, called Hyperledger Besu.
These smart contract were tested using Remix, an online IDE for Solidity running deployed contracts, and were built with the goal of using as little gas as possible per-function or-transaction consumption.

Finally, before showing the actual results, it is important to give an additional, basic, context of Etheremum and Gas:

- Ethereum is a platform powered by blockchain technology that is best known for its native cryptocurrency, called ETH, or simply ethereum.
- Gas refers, on the Ethereum blockchain, to the cost necessary to perform a transaction on the network. Miners set the price of gas based on supply and demand for the computational power of the network needed to process smart contracts and other transactions. The actual gas "price" is calculated in an unit called Gwei. One Gwei is equal to 0.000000001 ETH.

5.11.1 First test: one actor, more operations

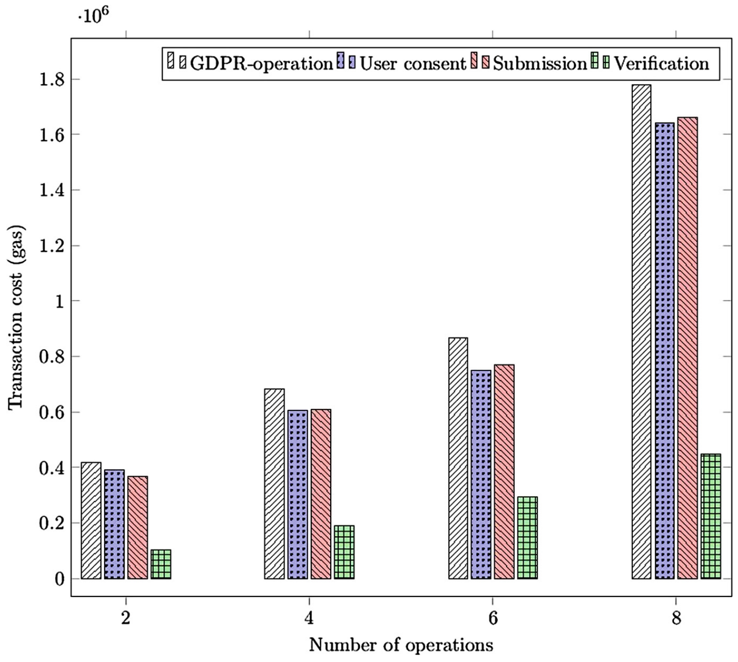


Figure 5.26: Consumption tests on the four smart-contracts. Image by [3]

This first test, while having just only one IoT device (the Controller, in our study case), let the **number of operations vary** from 2 to 8. Each operation contains a number of personal data, randomly selected from 1 to 5.
The amount of utilized gas gradually grows as the number of operations increases, as shown in the bar chart.
For istance, in the case of the user consent smart-contract, it shows a consumption of 390673 gas when there are 2 operations, and this consuption grows to 1641859 gas when there are 8 operations (so it can be noted that, in this case, an increase of 4 times the number of operations led to an increase of 4.2026 times the number of gas, so it is "almost" linear).
The most portion of cost should be paid for the transactions implemented in GDPR-operation contract, since they had the greatest number of opcodes compared with the transactions of other contracts.

5.11.2 Second test: more actors, one operation

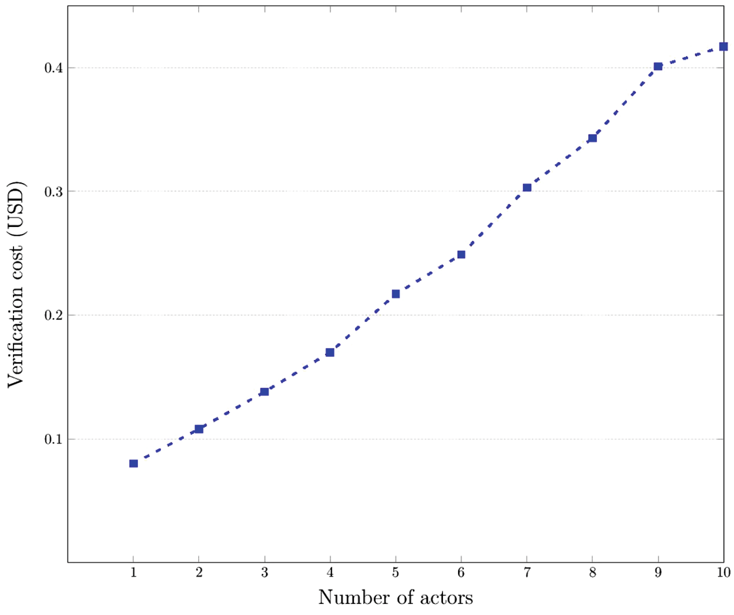


Figure 5.27: Test 2.1 by [3]

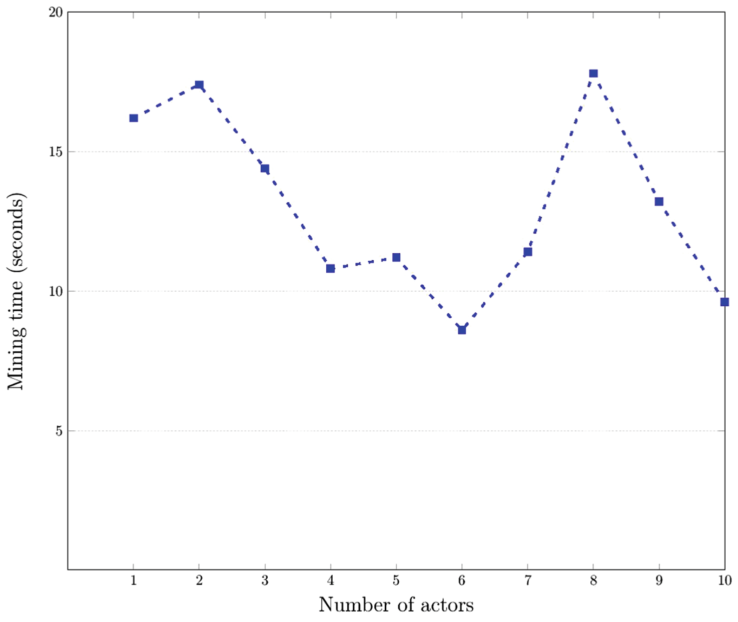


Figure 5.28: Test 2.2 by [3]

The second test, let the **number of actors vary** from 1 to 10 (while in the first test there was just 1 actor), and each actor executes just one operation. As in the first test, each operation contains a number of personal data, randomly selected from 1 to 5. For the given number of actors (again, going from 1 to 10), one of them is supposed to violate user consent.

Test 2.1 shows the relationship between the number of actors and the cost for detecting the violation through verification contract, having in mind that, during their tests, the cost of gas was 3.3 Gwei.
As can be seen from the graph of test 2.1, the verification cost rises constantly as the number of actors grows. In fact, as the number of actors grows, the verification procedure becomes more complex, necessitating more storage capacity.

Test 2.2 shows that the mining time (in seconds) taking for executing the transactions of verification contract has not a direct relationship with the number of actors. Instead, the time needed depends more on the miner's interest in mining the verify function.

5.12 Future works

Below, are listed some future works, useful to improve and implement this initial solution:

- **Lightweight Digital Signature: actual applications**: deepening of existing algorithms of Lightweight Digital Signature, looking at performance and safety, as the ones exploiting complex numbers instead of exponentiation;
- **Deepen privacy compliance in non-European countries (outside the Binding Corporate Rules)**: what would happen, if our blockchain will need to work in countries that do not have adequate privacy regulations?
- **Chameleon Hash**: effective application: deepening of existing algorithms of Chameleon Hash, with a particolar look to performance and safety;
- **Implement the proposed solutions in the blockchain**
  - How much gas is it using (on the Ethereum blockchain, gas refers to the cost necessary to perform a transaction on the network)?
  - Can the performance be improved?
  - Decentralize the model: modify the final model in order to allow all the nodes of the blockchain (the actors of the different companies) to add

transactions to the blockchain, unlike the current state where, due to the fact that we are in an early stage of development of the blockchain, all the data, as required, must pass through the central node of the Stellantis' DB. By doing so, more robustness and scalability would be achieved, deleting this many-to-one traffic flows leading to a possible single point of failure and information delay problems.

- **The controller is a potential weak physical point (point of failure)**: No details on the security of the Controller (as a physical sensor) are provided, what would be the consequences of a possible tampering of it? Can the security of this point be improved? At the moment, a successful attack to the controller, even if very difficult and unlikely to happen, would result in the possibility for this, to enter untrue transactions related to that given battery. The problem, however, would not affect other sectors and other batteries, and could be easily detectable given that we are talking about physical objects, continuously under control in the various sectors of the shipment.

Bibliography

[1] BINANCE ACCADEMY. La proof of authority spiegata. https://academy.binance.com/it/articles/proof-of-authority-explained. Accessed: 2022-02-23.

[2] Giuseppe Ateniese, Bernardo Magri, Daniele Venturi, and Ewerton Andrade. Redactable blockchain – or – rewriting history in bitcoin and friends. In *2017 IEEE European Symposium on Security and Privacy (EuroS P)*, pages 111–126, 2017.

[3] Masoud Barati and Omer Rana. *Enhancing User Privacy in IoT: Integration of GDPR and Blockchain*, pages 322–335. https://www.researchgate.net/publication/338813138, 01 2020.

[4] Fangfang Dai, Yue Shi, Nan Meng, Liang Wei, and Zhiguo Ye. From bitcoin to cybersecurity: A comparative study of blockchain application and security issues. In *2017 4th International Conference on Systems and Informatics (ICSAI)*, pages 975–979, 2017.

[5] Daniel. How do blockchain networks validate data? https://ico.li/blockchain-validate-data/. Accessed: 2021-10-30.

[6] Ashutosh Dhar Dwivedi, Gautam Srivastava, Shalini Dhar, and Rajani Singh. A decentralized privacy-preserving healthcare blockchain for iot. *Sensors*, 19(2), 2019.

[7] Danielle Enwood. Zero-knowledge proofs – a powerful addition to blockchain. https://blockheadtechnologies.com/zero-knowledge-proofs-a-powerful-addition-to-blockchain/. Accessed: 2021-11-20.

[8] M. Finck. Blockchains and data protection in the european union. *SSRN Electronic Journal*, 2017.

[9] L.-D. Ibáñez H. Fryer and E. Simperl. Attaching semantic metadata to cryptocurrency transactions. *in Workshop on Decentralising the Semantic Web, 2017*, 2017.

[10] Rosita Galiandro. Cybersecurity per la blockchain e la blockchain per la cybersecurity. `https://www.ictsecuritymagazine.com/articoli/cybersecurity-per-la-blockchain-e-la-blockchain-per-la-cybersecurity-3-3/`. Accessed: 2021-10-30.

[11] ADAM HAYES. Blockchain explained. `https://www.investopedia.com/terms/b/blockchain.asp`. Accessed: 2022-02-23.

[12] IBM. What are smart contracts on blockchain? `https://www.ibm.com/topics/smart-contracts`. Accessed: 2021-12-20.

[13] Luis-Daniel Ibáñez, Kieron O'Hara, and Elena Simperl. On blockchains and the general data protection regulation. *https://www.researchgate.net/*, 07 2018.

[14] Abradat Kamalpour. Privacy laws and blockchain. `https://www.cityam.com/privacy-laws-and-blockchain/`. Accessed: 2021-10-30.

[15] Georgios Konstantopoulos. Understanding blockchain fundamentals, part 2: Proof of work & proof of stake. `https://medium.com/loom-network/understanding-blockchain-fundamentals-part-2-proof-of-work-proof-of-stake-b6ae907c7edb`. Accessed: 2021-11-15.

[16] Ying-Chang Liang. *Blockchain for Dynamic Spectrum Management*, pages 121–146. ResearchGate, 01 2020.

[17] Lucas Nuzzi. Monero becomes bulletproof. `https://medium.com/digitalassetresearch/monero-becomes-bulletproof-f98c6408babf`. Accessed: 2021-11-22.

[18] GUIDO PERBOLI, VITTORIO CAPOCASALE, and STEFANO MUSSO. Quadrans performance evaluation.

[19] Guido Perboli, Stefano Musso, Mascolo Julien, Vittorio Capocasale, and Mariangela Rosano. Integration of blockchain in supply chain and logistics networks.

[20] JONATHAN ROUACH. Data privacy is forever changed. zero-knowledge proofs are enterprise's solution. `https://forkast.news/data-privacy-is-forever-changed-zero-knowledge-proofs-are-enterprises-solution-opinion/`. Accessed: 2021-11-20.

[21] Damiano Sartori. Redactable blockchain : how to change the immutable and the consequences of doing so, August 2020.

[22] European Parliamentary Research Service. Blockchain and the general data protection regulation. `https://www.europarl.europa.eu/RegData/etudes/STUD/2019/634445/EPRS_STU(2019)634445_EN.pdf`. Accessed: 2021-10-30.

[23] TOSHENDRA KUMAR SHARMA. Permissioned and permissionless blockchains: A comprehensive guide. `https://www.blockchain-council.org/blockchain/permissioned-and-permissionless-blockchains-a-comprehensive-guide/`. Accessed: 2021-11-07.

[24] TecraCoin. what is genesis block and why genesis block is needed? `https://tecracoin.medium.com/what-is-genesis-block-and-why-genesis-block-is-needed-1b37d4b75e43`. Accessed: 2021-12-18.

[25] Stanford University. Short proofs for confidential transactions and more. `https://crypto.stanford.edu/bulletproofs/`. Accessed: 2021-11-22.

[26] Ethereum Wiki. Proof of stake faqs. `https://eth.wiki/en/concepts/proof-of-stake-faqs`. Accessed: 2021-11-15.

[27] Wikipedia contributors. Homomorphic encryption — Wikipedia, the free encyclopedia. `https://en.wikipedia.org/w/index.php?title=Homomorphic_encryption&oldid=1055930805`, 2021. [Online; accessed 21-November-2021].

[28] Wikipedia contributors. Ring signature — Wikipedia, the free encyclopedia, 2021. [Online; accessed 18-February-2022].

[29] Wikipedia contributors. Speck (cipher) — Wikipedia, the free encyclopedia, 2021. [Online; accessed 6-February-2022].

[30] Zibin Zheng, Hong-Ning Dai, Mingdong Tang, and Xiangping Chen, editors. *Blockchain and Trustworthy Systems - First International Conference, BlockSys 2019, Guangzhou, China, December 7-8, 2019, Proceedings*, volume 1156 of *Communications in Computer and Information Science*. Springer, 2020.

[31] Valentin Zieglmeier and Gabriel Loyola Daiqui. Gdpr-compliant use of blockchain for secure usage logs. *Evaluation and Assessment in Software Engineering*, Jun 2021.