VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belgavi-590010

Sir M. Visvesvaraya Institute of Technology
Bengaluru, Karnataka-562157

Mini Project Report on

“Blockchain enabled Anti-Counterfeiting Suite”

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

For the Academic Year 2020-2021.

Submitted by,
Aditya Mondal 1MV18EC004

Sir M. Visvesvaraya Institute of Technology

Bengaluru, Karnataka-562157

CERTIFICATE

Mr./Ms. ***ADITYA MONDAL*** , USN ***1MV18EC004*** , a bonafide student of ***Sir M Visvesvaraya Institute of Technology*** in partial fulfillment for the award of Bachelor of Engineering / Bachelor of Technology in ***Electronics & Communication*** of the Visveswaraiah Technological University, Belgaum during the year ***2020-2021*** . It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the Report deposited in the departmental library.

The project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said Degree.

Signature of the Guide: Smt. Rajeshwari KN	Signature of the HOD: Dr. R Sundarguru	Signature of the Principal: Dr. V.R. Manjunath

ACKNOWLEDGEMENT

It is our earnest duty to express our thanks to all those who contributed directly or indirectly to our project.

Firstly, we would like to thank **Sir M. Visveswaraya Institute of Technology** & Department of Electronics & Communication for giving us an opportunity. We would like to thank our Teacher & Guide Prof. Smt. Rajeshwari KN, who initiated us to complete this project & guided us timely.

Last but not least I express my gratitude to all staff members & all our friends for their excellent suggestions & Coordination. There might be some problems or Bugs & additional requirements with this system. In future these problems will be corrected accordingly. For this your valuable suggestions are most welcomed.

We Once again, we thank you all for your cooperation

ABSTRACT

For more than a decade now, radio frequency identification (RFID) technology has been quite effective in providing anti-counterfeits measures in the supply chain. However, the genuineness of RFID tags cannot be guaranteed in the post supply chain, since these tags can be rather easily cloned in the public space. Here, we explore a novel approach of RFID-attached products to detect anti-counterfeits and explore a solution that can be used in the post supply chain. For this purpose, we leverage the idea of Bitcoin's blockchain that anyone can check the proof of possession of balance. With the proposed solution, a customer can reject the purchase of counterfeits even with genuine RFID tag information, if the seller does not possess their ownership. We implement a proof-of-concept experimental System employing a blockchain-based Decentralized Ethereum, and evaluate its cost performance.

Table of Contents

Blockchain enabled Anti-Counterfeiting Suite

By Aditya Mondal

INTRODUCTION

Counterfeiting products, such as branded goods, is one of the most important and difficult issues to deal with in national/international markets. This has been recognized for more than a decade now, as the OECD (Organisation for Economic Co-operation and Development) announced in 2007 that the counterfeits in international trade could amount to about US$250 billion. Because of the rapid development of e-and i-commerce, clearly there exists an urgent demand to develop anti-counterfeits systems. On the other hand, for over a decade, RFID (Radio Frequency IDentification) as a key technology in the IoT (Internet of Things) world, has received a lot of attention since it can be used to detect counterfeits which are inserted into the supply chain. In the RFID-enabled supply chain, an EPC (Electronic Product Code) is assigned to each product and is written into an RFID tag. Such tag-attached products are shipped from manufacturers to the supply chain parties. During the transportation process, each involved party interrogates RFID tags and adds extra evidence data into tags. In this way, the next party can check whether or not the products have passed through the legitimate supply chain. If any inconsistency is found, such products may be considered as counterfeits. However, once the products reach the end of the supply chain and are displayed in retail stores, their genuineness is no longer guaranteed, as anyone who has an RFID reader can interrogate and clone tags’ information. Therefore, it is important to develop anti-counterfeits systems that work even when the tag’s information is cloned in the post supply chain. In this project, we explore a novel approach which makes the efforts of counterfeiters to clone genuine tags redundant since they cannot prove the possession of products on this system. For this purpose, we leverage the idea of Bitcoin, a decentralized cryptocurrency system in which the possession of the user’s balance can be proven in the public ledger referred to as blockchain. In particular, by borrowing the ideas first presented in the ‘‘proof of possession of balance’’ used in Bitcoin, we introduce here the concept of ‘‘proof of possession of products’’. Furthermore, several issues must first be identified and then be addressed for the successful realization of a solution with blockchain technology. For example, only the legitimate manufacturers can claim the first ownership of their own products. To comply with such requirements, here we propose a novel blockchain-based solution for anti counterfeits which works very effectively for the post supply chain. Firstly, the overall practical system requirements are identified. Then, we introduce a full-fledged protocol that enables each party, including supply chain partners and customers, to transfer and prove the ownership of RFID tag attached products. An important advantage of the proposed solution is that customers can reject the purchase of counterfeits, even with a genuine EPC, under the condition that the seller does not possess their ownership. Based on the proposed protocol, a proof-of-concept experimental system has been implemented on the Ethereum platform.

Problem statement

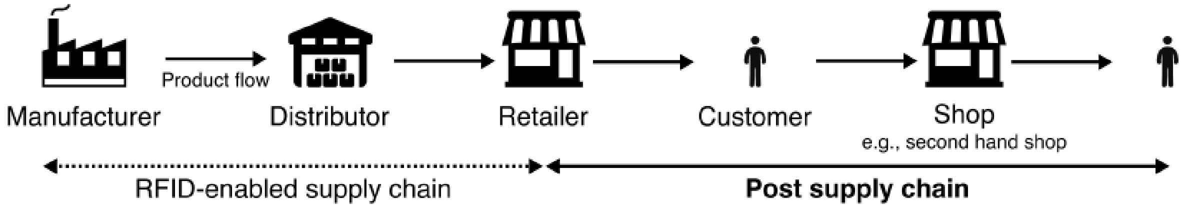

Fig 1

For more than a decade now, radio frequency identification (RFID) technology has been quite effective in providing anti-counterfeits measures in the supply chain. However, the genuineness of RFID tags cannot be guaranteed in the post supply chain, since these tags can be rather easily cloned in the public space. Here, we explore a novel approach of RFID-attached products to detect anti-counterfeits and explore a solution that can be used in the post supply chain. For this purpose, we leverage the idea of Bitcoin's blockchain that anyone can check the proof of possession of balance. With the proposed solution, a customer can reject the purchase of counterfeits even with genuine RFID tag information, if the seller does not possess their ownership. We implement a proof-of-concept experimental system employing a blockchain-based decentralized application platform, Ethereum, and evaluate its cost performance.

Literature survey

Based on Previous research done in A Blockchain-Based Application System for Product Anti-Counterfeiting by Jinhua Ma, Shih-Ya Lin, Xin Chen, Hung-Min Sun, Yeh-Cheng Chen, (Graduate Student Member, IEEE) and Huaxiong Wang[1], which was published on IEEE Access (Volume 8) on Feb 2020, we lay our ground work and model for the Project. In addition to this, How Luxury Brands Can Beat Counterfeiters by Roberto Fontana, Stéphane J.G. Girod, Martin Králik published in the Harvard Business Review[2] is also referred for consideration and inspiration. And lastly, for the development of the Project Ethereum Developer's Documentation[3] is extensively referred to.

In this section, firstly, the system model is introduced. Then, an overview of the previously published papers and the motivation behind this will be presented. Thirdly, the fundamentals of blockchain and Bitcoin technologies, and how these can be considered within the blockchain operation are explained.

A. SYSTEM MODEL

Fig. 1 illustrates the typical product flow consisting of two chains, namely the RFID-enabled and post supply chains. The first one is typically composed of three parties, i.e., manufacturers, distributors, and retailers. A manufacturer creates, composes, and ships products to the distributors while they decompose the received products and ship them further to the retailers. In the post supply chain, retailers stock

[1] https://ieeexplore.ieee.org/document/8985337
[2] https://hbr.org/2019/05/how-luxury-brands-can-beat-counterfeiters
[3] https://ethereum.org/en/developers/docs/

and sell their products to customers who in turn may resell them, e.g. at a second hand shop or over the Internet. We consider that each supply chain party has EPCglobal Class 1 Gen 2 (C1G2) compatible RFID readers2 A manufacturer assigns an EPC to every product and EPCs are written into tags attached to the products so that any party can recognize products when they arrive. It is also assumed that every party has an Internet connection via computers and/or smartphones/tablets.

B. PREVIOUS WORK AND MOTIVATION

For over a decade now, RFID technology has been integrated into the supply chain for anti-counterfeits. The first systematic RFID-based approach for anti-counterfeits in the food and drug industry was proposed by the FDA (Food and Drug Administration) in the USA. In their proposal, each supply chain party is equipped with RFID readers and keeps track of shipping and receiving events for each product. In this way, the supply chain parties have the ability to track and trace the product flow of products. However, such an approach is vulnerable against cloned tags. Specifically, once RFID tags attached to the genuine products are copied by an attacker, counterfeits with cloned RFID tags can be inserted in the supply chains. In this way, counterfeits with cloned tags cannot be identified by the aforementioned track and trace approach. So far, the research efforts dealing with this problem can be classified into two categories. The first one involves the development of secure tag distribution schemes so that an attacker cannot copy the contents of tags in the supply chains. Among these works, perhaps the most important one is a secret sharing based key distribution scheme. In this scheme, the tag’s information is encrypted with a symmetric encryption scheme and an encryption key is split into multiple shares by a secret sharing scheme. The secret sharing scheme realizes that one can extract the key if he/she can obtain more than a certain amount of unique shares. An encrypted EPC and the shares of a key are written into a tag on the product. After receiving products, an authorized partner interrogates tags and recovers the key from a sufficient number of shares. Then, the EPCs are decrypted with the extracted key. The second category deals with cloned tags detection schemes in the RFID-enabled supply chains. For example, when a tag arrives at a supply chain party, it writes an arrival evidence to the next available position in the tag memory. Then, the party requests a detector who can obtain information about a supply chain to check whether or not the information in the tag is valid. By doing so, a detector can identify counterfeits if any inconsistency is found, i.e., written positions are wrong and/or written words are not matched. However, once the products are for sale in retail stores, i.e., in the post supply chain scenario, it is not feasible to execute any such track and trace methods nor secure EPC distribution schemes. In this case, none of the currently employed track and trace methods can guarantee the tag attached the product's genuineness because they leverage the tag’s secret information. In addition, it is obvious that once they are displayed in public, any secure EPC distribution does not make much sense. Hence, EPCs are not assured any more to be unique and genuine. Thus, it is crucial to consider the detection of counterfeits even in the post supply chain. For this, an anti-counterfeits scheme allows a customer to check the legitimacy of products. However, this approach requires a special computational tag for each product and thus publicly available off-the-shelf tags cannot be used. Moreover, if a customer sells such products to a second hand shop or at the auction, it will also require customers to register their certificates at PKI (Public Key Infrastructure), which is not realistic. Motivated by the above, our basic idea here is to introduce here the concept of proving ‘‘the possession of products’’ and designing a novel POMS which manages and tracks the possession of products starting from their manufacturers to the current owners. With such a scheme, counterfeits may be detected if a party cannot prove the possession of claimed products. For example, let’s assume that a customer wants to buy a branded product through a second hand shop. A genuine EPC is attached to this selling product but actually , in reality, it is a counterfeit. In this case, it is impossible for the customers to check whether or not the seller possesses the ownership of the claimed EPC. To check this, we need to verify if (i) The initial owner of the product with the claimed EPC must be its manufacturer, and (ii) The current owner of the product with the claimed EPC must be the seller. The obvious and straight-forward way to validate the two conditions is that manufacturers construct a large-scale system which manages the ownership of their products. However, this is a very complex and costly procedure since it must also be secured against attacks, e.g. DDoS (Distributed Denial of Service) attacks. Furthermore, such a system typically requires the involvement of many parties and rather complex procedures, including the consumers to register their identification, that is, dealing with sensitive data, e.g. ID/password pairs. Therefore, a more efficiently operating POMS is required for dealing with such issues and at the same time being compatible with the RFID-enabled supply chain.

C. BLOCKCHAIN TECHNOLOGY

To realize such a POMS, we leverage the idea of Bitcoin, a decentralized cryptocurrency system in which the possession of a user's balance can be proven in the public ledger referred to as blockchain. Specifically, we replace its concept of ‘‘proof of possession of balance’’ with an equivalent concept which we will refer to as ‘‘proof of possession of products’’. A few startup companies have just started (or soon plan to start) services with the blockchain for managing the genuineness of products. For example, Everledger appears to be the most successful service specific for diamonds. Everledger offers a permanent ledger for diamond certification and related transaction history. This special ledger is used for verification for insurance companies, owners, claimants and law enforcement. Another example is Blockverify which appears to use the blockchain technology for pharmaceuticals and luxury items. However, no details whatsoever are available publicly about the protocol of such commercial services, meaning that its security and privacy issues cannot be reviewed. Nevertheless, it is clearly of importance to propose a rigorous and transparent protocol for blockchain-based POMS. In order to do so, the overview of the blockchain technology will be presented next and then our POMS will be described in detail in the next section.

1) BITCOIN AND BLOCKCHAIN

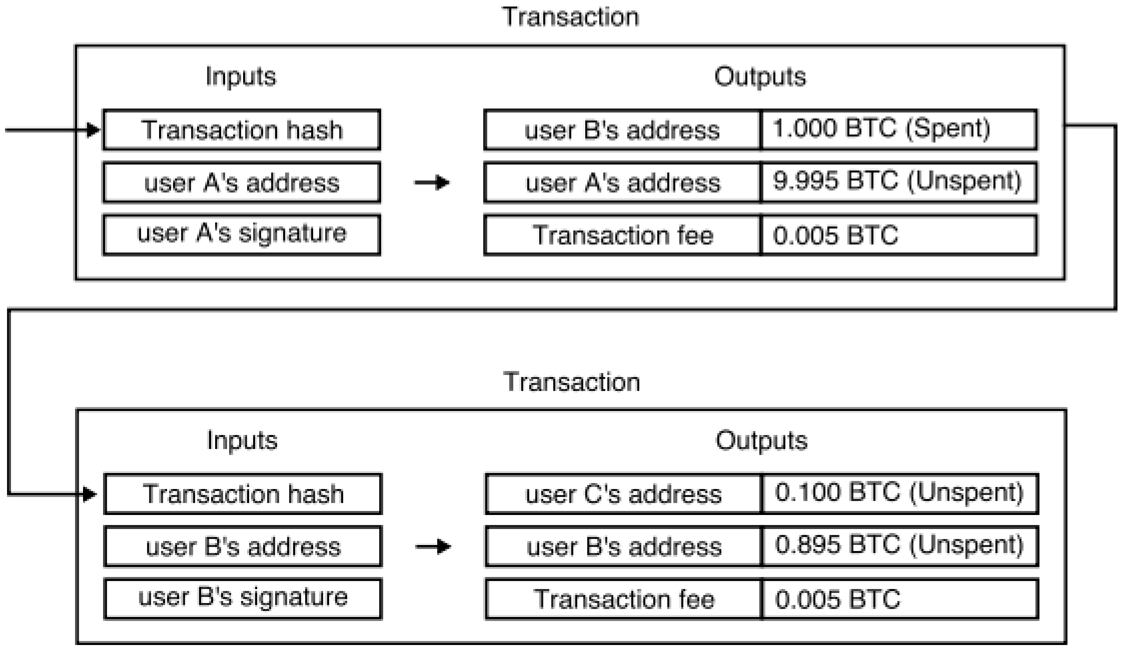

Fig 2

We first focus on Bitcoin, a decentralized cryptocurrency where any trusted authority, e.g. a bank, does not exist, because Bitcoin is the first successful system using blockchain technology. With the blockchain, Bitcoin allows the users to prove the ownership of their balance without any authentication and a centralized authority. It is useful first to briefly explain how a user transfers Bitcoin to another user. In Bitcoin, a data structure referred to as a transaction is used to identify the amount to be transferred, sender(s), and recipient(s). Fig. 2 illustrates an example of such Bitcoin transactions, which presents the detailed procedure of user A transferring his/her Bitcoin to user B and then user B transferring his/her Bitcoin to user C, as indicated in the first transaction in Fig. 2. For user A to transfer his/her Bitcoin to user B, user A’s address calculated from A’s public key, the source of Bitcoin (unspent balance user A owns), and its signature calculated by user A’s private key are specified in the input part of a transaction. At its output part, user A specifies the amount to be transferred and user B’s address, which is calculated from user B’s public key. User A also specifies his/her own address or possibly his/her newly created address owned by user A to receive the change. As illustrated in the second transaction in Fig. 2, when user B transfers Bitcoin to user C, user B follows the same procedure that user A did. It is well-known that in Bitcoin, any user can check every transaction. Hence, if the same address is used for several transactions, this could be a problem with the privacy issue. To reduce the effects of this privacy issue, it is recommended for users to generate a fresh address for every transaction. Since it is a complex task to manage multiple addresses, an application called wallet is used to efficiently store and generate public/private key pairs for every transaction. Since Bitcoin is decentralized, a consensus algorithm is necessary so that every user agrees on the order of transactions. Without a consensus, any malicious user might try to use an already spent balance twice, which is so-called double-spend. Bitcoin achieves (probabilistic) consensus by introducing the blockchain concept, which is an ever growing chain of blocks that contain approved transactions. Fig. 3 illustrates a sequence of blocks used to construct a blockchain. A block is created by imposing potential block creators, referred to as miners, to solve the computational puzzle that is difficult to solve but is easy to check. More specifically, miners are required to find a nonce (number used once) that satisfies its (SHA-256) hash value together with a reference to the previous block and a set of the unapproved transactions lowers the target value. Since the previous block is required to generate the next block, this creates an ever-growing chain

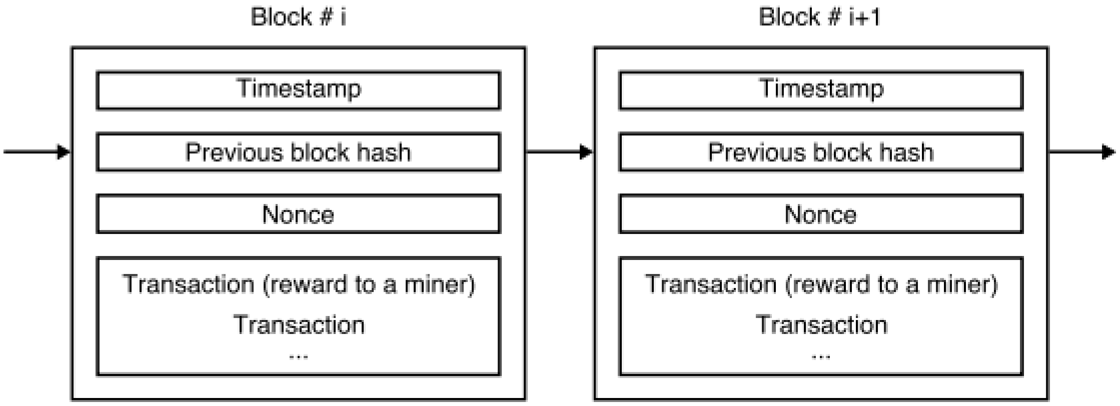

Fig 3

of blocks, i.e., blockchain. In order for the miners to follow such protocol, the following incentive mechanism is adopted: The first solver of the block acquires the newly minted Bitcoin as a reward, and in addition, all transaction fees which are included in the block. Although such a blockchain-based decentralized consensus mechanism is firstly used for ‘‘currency’’, it can be extended to other applications that need to be publicly verifiable. Clearly, POMS is such an application as it can be realized based on the blockchain by substituting the concept of ‘‘currency’’ with that of a ‘‘product’’. This notion has been already embodied as a CC (Colored Coin) protocol in that any Bitcoin user can issue its own currency. For example, a CC may represent an asset or a property, e.g. a car or a house. Most CC protocols were built by leveraging Bitcoin’s scripting function, in which a sender can specify rules to send a Bitcoin. In another approach, it was suggested to construct a container tracking system with a CC protocol by issuing and exchanging ‘‘I have a container’’ and ‘‘I received a container’’, tokens originally issued by manufacturers and other parties. However, it should be emphasized that the CC protocol cannot be used for the POMS because it lacks the operating requirements necessary for its successful implementation. Although these requirements will be presented with details in Section III-A, it is worthwhile mentioning here that for POMS it is necessary to provide incentives to the cooperative parties, so that the parties follow the protocol. Unfortunately, it is not possible to fulfill such a requirement because of its rather simple CC protocol available on Bitcoin. Therefore, to implement a well functioning POMS, we must choose another blockchain based consensus platform which can support the execution of any code. Ethereum is one such platform that fulfills this condition, and thus its operation will be described next.

2) ETHEREUM

Ethereum is a blockchain-based decentralized cryptocurrency where any code execution is possible. To enable code execution, memory storage is required on accounts. For this reason, Ethereum has two types of accounts: CA (Contract Account) and EOA (Externally Owned Account). On the one hand, an EOA manages its own balance. On the other hand, a CA has its own code as well as storage and executes the code when a message is received from another CA or EOA. To write a code for a CA, a scripting language called Solidity is typically used. The code written in Solidity is compiled into the stack-based programming language so that a CA can execute it. Since any algorithm can be described on Ethereum with Solidity, the POMS for anti-counterfeits can be also implemented on Ethereum. Supporting the programming language means that an attacker can make miners keep computing meaningless codes, e.g. causing an infinite loop, wasting computing resources and energy. To avoid this attack, any code execution that changes the storage requires a sender to contract a sufficient amount of ‘‘gas’’ according to procedures, e.g. data amount and the number of computational steps. If ‘‘gas’’ runs out during the code execution, the state will be reverted to the original one, but the cost for ‘‘gas’’ is not returned to the sender.

Objective

In this project, we explore a novel approach which makes the efforts of counterfeiters to clone genuine tags redundant since they cannot prove the possession of products on this system. For this purpose, we leverage the idea of Bitcoin, a decentralized cryptocurrency system in which the possession of the user's balance can be proven in the public ledger referred to as blockchain. In particular, by borrowing the ideas first presented in the ‘‘proof of possession of balance’’ used in Bitcoin, we introduce here the concept of ‘‘proof of possession of products’’. For this to happen we want that,

1) Only the legitimate manufacturers are able to claim the initial ownership (origin) of products (EPCs or Electronic Product Codes);

2) Each manufacturer can declare only their own products;

3) The events ‘‘Shipped’’ and ‘‘Received’’ can be separated;

4) A manufacturer must give some incentive to each party who follows the POMS protocol.

Methodology Followed

In this section, an overview of the proposed POMS will be presented. In particular, the key system requirements will be first outlined followed by the pseudo-codes of the implemented contracts in Ethereum. Next, the details of the algorithmic procedures between all parties necessary for the realization of the proposed POMS will be identified. As it will become apparent in the last part of this section, the main advantage of the proposed POMS is that it makes tags cloning meaningless since even if tags are cloned by the counterfeiters, they cannot prove the ownership of the products.

A. POMS OPERATIONAL REQUIREMENTS

Before describing the POMS protocols in detail, we present the key requirements and explain their necessity for the proper operation of the proposed product ownership proof system.

1) Only the legitimate manufacturers are able to claim the initial ownership (origin) of products (EPCs or Electronic Product Codes);
2) Each manufacturer can declare only their own products;
3) The events ‘‘Shipped’’ and ‘‘Received’’ can be separated;
4) A manufacturer must give some incentive to each party who follows the POMS protocol.

The first requirement is necessary to avoid any unauthorized parties including counterfeiters from illegally issuing the ownership of products. The second one is also required, since without it, the following case could be possible: A manufacturer, M1, claims the initial ownership of manufacturer, M2's, products. This can be avoided by leveraging the company prefix specified in the EPC format of SGTIN-96, which is the 96 bits EPC format to identify products. Table 1 shows the message format of SGTIN-96. As it can be seen from this table, SGTIN-96 includes the company prefix which identifies its manufacturer. In reality, each manufacturer may enroll its own company prefix in GS1 in the RFID-enabled supply chain. By leveraging this, manufacturers can claim the initial ownership of only their own EPCs whose company prefix is enrolled in GS1. Regarding the third requirement, the reason why we divide the ownership transfer process is to avoid the situation where the current owner sends the product to the recipient while it does not arrive at its destination. In this case, if only a function that simply transfers the ownership of a product were implemented, the following undesired case might occur: The ownership is transferred, however, the product itself is not arrived at the recipient. The last requirement will enable the proper operation of the overall system, because every party except for manufacturers has to pay fees to issue a contract which transfers the ownership of products. Hence, without incentive mechanisms, the non-manufacturer parties will make a loss and eventually have no motivation to follow the POMS protocol.

B. IMPLEMENTED CONTRACTS

To satisfy the above requirements, we have implemented two contracts, namely MM (ManufacturersManager) and PM (ProductsManager). On the one hand, MM offers functions for managing the information of manufacturers, e.g. enrollment of a company prefix registered in GS1 and manufacturer's address. On the other hand, PM is operated by each manufacturer and offers functions to manage the information of products, e.g. enrollment of a new product and ownership transfer. In contrast to PM, in MM we assume the existence of an administrator who manages the manufacturers' information. To avoid impersonation, only the administrators can modify any manufacturer's information. One of the administrative candidates is GS1, because it manages company prefixes. Although this may break the assumption of a fully decentralized system, it is inevitable in order to avoid impersonators, e.g. counterfeiters, from registering themselves as if they are legitimate manufacturers. Actually, it might be possible to make our MM decentralized by leveraging the notion of Namecoin [29]. Namecoin is a decentralized domain name system and avoids ‘‘massive’’ registration by imposing a cost for enrollment. However, if our MM might be constructed by introducing a registration fee like Namecoin, there could be a chance for counterfeiters to illegally register themselves as genuine companies by paying the appropriate fees. However, it still might be possible to make MM decentralized. This is one of the open questions regarding the blockchain-based POMS.

1) MANUFACTURERS MANAGER (MM)

MM is mainly composed of two functions:

1) enrolling the manufacturer’s information in the blockchain and
2) checking the authorship for a requesting manufacturer to enroll the product’s EPC.

```
● Inputs:
        Manufacturer M’s address (AM ), company prefix (companyPrefix), name
        (companyName), and valid duration (validDuration)
● if
        the sender of this transaction is an Admin (like GS1)
  then
        Enroll manufacturer M’s information, i.e., AM , companyPrefix,
        companyName, and validDurationon the blockchain
● else
        Do nothing
● end if
```

Algo 1 : enrollManufacturer() Enrolling a Manufacturer’s Information

Algo. 1 shows the pseudocode of enrollManufacturer(), which enrolls the manufacturer's information required when its product is stored in the blockchain. Since our POMS requires that only one administrator, e.g. GS1, can enroll the manufacturer's information, this condition is checked at step 2. If it is True, then the admin enrolls the manufacturer's information in the blockchain.

```
● Inputs:
        The message sender's address (Amsg) and EPC
● Output: A boolean (True or False)
         companyPrefixmsg ← Get the message sender's companyPrefix through
                               the blockchain
● if companyPrefix described in the claimed EPC is the same with
  companyPrefixmsg then
        Return True
● else
        Return False
● end if
```

Algo 2: Pseudo-Code of checkAuthorship(), Which Checks Whether or Not a Message Sender Possesses the Authorship of a Claimed EPC

Algo. 2 shows the pseudo-code of checkAuthorship(), which checks whether or not a message sender possesses the authorship of a claimed EPC. This function is invoked when a message sender (possibly a manufacturer M) claims to be the initial owner of its product. In this case, firstly the message sender's company prefix is retrieved by querying its address in the blockchain. Then, it is compared with the company prefix extracted from the EPC available in the argument. If these prefixes match, it can be confirmed that the message sender is actually the enrolled manufacturer M and the function returns True. Otherwise, it simply returns False.

2) PRODUCTS MANAGER (PM)

In contrast to MM, the contract PM is created by each manufacturer and consists of four main functions:

1) **enrollProduct():** Invoked when a manufacturer M first enrolls its own product specified by unique EPC and claims its initial ownership;
2) **shipProduct():** Invoked when a current owner parts with a product and specifies the recipient;
3) **receiveProduct():** Invoked by the new owner to successfully transfer its ownership when a product is successfully received;
4) **getCurrentOwner():** Returns the current owner's address.

```
● Inputs: The PM's address (Amsg) and EPC to be enrolled
● if the message sender has the authorship to claim the initial owner of EPC
  through checkAuthorship() in PM then
        Specify EPC's status as Owned
        Specify EPC's owner as Amsg
        Specify EPC's number of transfer (nTransferred) as 0
        Enroll these information on the blockchain
● else
        Do nothing
● end if
```

Algo 3: Pseudo-Code of enrollProduct() for Enrolling a Product on the Blockchain

Algo. 3 shows the pseudocode of enrollProduct(), which enrolls the manufacturer's information required when its product is stored on the blockchain. Since our POMS restricts that only a single administrator, e.g. GS1, can enroll the manufacturer's information, this condition is checked at step 2. If found to be True, then the administrator enrolls the manufacturer’s information in the blockchain. Note that we assume that an administrator has manually checked that the manufacturer’s information is legitimate before sending this transaction. This notion is analogous to the enrollment of a certificate in PKI [30].

```
● Inputs: The message sender’s address (Amsg), the recipient’s address (Arec), and EPC to be transferred
● if The product with EPC really exists, EPC’s status is Owned, and the message sender is the owner of EPC then
        Specify EPC’s recipient as Arec
        Specify EPC’s status as Shipped
        Enroll these information on the blockchain
● else
        Do nothing
● end if
```

Algo 4: Pseudo-Code of shipProduct() Called When a Product Is Just Left From the Current Owner

Alg. 4 shows the pseudo-code of shipProduct() which is for the current owner to transfer the product to the next owner. At first, the function checks that the given EPC exists on the blockchain and the sender of the message actually has the ownership of the EPC. Then, if this is True, shipProduct() specifies the “recipient” as the next recipient’s address, Arec and EPC’s status as Shipped. Note that, at this point in time, the POMS does not change the ownership of the product because of the possibility that this might get lost during the transportation process.

```
● Inputs: EPC that a new owner received
● if the message sender is the recipient of the EPC specified by the current owner.
  then
          Specify the EPC’s owner as Amsg
          Specify the EPC’s status as Owned
● Update nTransferred as nTransferred + 1
          if nTransferred <= MAXTRANSFER then
                  Manufacturer M sends incentive as specified by transferReward
          end if
● else
          Do nothing.
● end if
```

Algo 5: Pseudo-Code of receiveProduct() Invoked by a New Owner Who Received a Product

Alg. 5 describes the receiveProduct() which is for the receiver to confirm the arrival of the product. The function checks that the claimed EPC is specified by the current owner and that the status of the EPC is Shipped. If this is True, the ownership is successfully transferred to the message sender’s address. In addition, the manufacturer of the product gives incentive, i.e., some ETH, to the message sender as a reward for obeying the protocol. Since Ethereum requires an execution fee for each transaction, when the current owner sends a product to the recipient, he/she might be reluctant to issue a transaction shipProduct(). To avoid this kind of situation, the following procedure is introduced. If the ownership transfer has been successfully completed, a financial reward transferReward is paid back to the previous owner by the product’s manufacturer. The reason why the manufacturer should pay such reward is that in this way counterfeits can be detected and thus identified thanks to their cooperation. It is noted that, in order to avoid the case where two parties repeatedly transfer back and forth to earn rewards, we specify a maximum number of transfers, referred to as MAXTRANSFER. Selecting appropriate values for transferReward and

MAXTRANSFER will depend on the actual investment made by the manufacturer for the implementation of POMS. However, such a topic is outside the scope of our current research, and thus it will not be considered further.

C. ALGORITHMIC PROCEDURES

Fig. 4 illustrates the detailed system model of the proposed POMS, where two groups of parties are identified. The first one belongs to the supply chain, that is, an administrator, such as GS1, manufacturers, distributors, and retailers. The second group belongs to the post supply chain parties, that is, second hand shops and consumers. Each party possesses Ethereum accounts and manages them with a wallet application operating on personal computers or smartphones/tablets. Next, the procedures on each party in (i) the supply chain and (ii) the post supply chain, will be presented.

```
● Input: EPC Output: the current owner's address of a product EPC
● if The product with EPC really exists then
          Return the current owner's address of a product EPC
● end if
```

Algo 6: Pseudo-Code of getCurrentOwner(), Which Returns the Current Owner's Address of a Product EPC

1) THE SUPPLY CHAIN

In the supply chain, the following five operational steps are taken.

1) Manufacturer M sends a transaction to enrollManufacturer() to a contract MM to enroll its own company prefix specified in EPCs and the company name. Note that this enrollment step requires authentication to avoid counterfeiters from illegally claiming non-authorized company prefix and name.
2) Manufacturer M manufactures Nproducts products and assigns a unique EPC, denoted by EPCi , for each product i where 1 ≤ i ≤ Nproducts. Simultaneously, to

claim the initial ownership of the products, Manufacturer M sends a transaction enrollProduct() to a contract PM to enroll Nproducts EPCs, that is, (EPC1, EPC2, · · · , EPCNproducts). Since off-the-shelf smartphones and tablets are not equipped with an RFID reader, it is certainly useful and user-friendly that an EPC is also written into a QR code [8] or an NFC (Near Field Communication) tag as proposed in [19].

3) After shipping Nproducts products, manufacturer M issues a transaction shipProduct() to PM for each product to inform distributor D that manufacturer M is now ready to transfer the ownership of the products.
4) When receiving products, distributor D reads the tags' EPCs and checks the genuineness of the EPC. Specifically, for each EPC, distributor D invokes getManufacturerAddress() with EPCs and obtains the manufacturer's address. Then, distributor D invokes getRecipient(), getProductStatus(), and getCurrent Owner() , which is shown in Alg. 6. Distributor D verifies the genuineness of the EPC and issues a transaction receiveProduct() to the contract PM with interrogated EPCs, if all of the following conditions are met:
    a) The distributor D's address is specified as the recipient;
    b) The product status is Shipped;
    c) The address of the current owner is that of a manufacturer M.
5) If distributor D further transfers products to other parties, i.e., retailers, the same procedure as in the above described step 3 is followed. Similarly, when any party receives products, a recipient follows the same procedure as in step 4.

2) THE POST SUPPLY CHAIN

In the post supply chain, the consumer will decide to buy a product after verifying that the seller actually possesses the ownership of the product. Hence, in contrast to the ‘‘supply chain situation’’, an extra step is required before the seller issues a transaction shipProduct(). In the following, we assume the situation where consumer C is about to buy a product from its current owner.

1) Consumer C obtains the EPC of the desired product and the current owner's address. Regarding EPC, the buyer can interrogate the EPC of the product via an RFID reader, QR code, or NFC when he/she is physically present in a shop. In contrast, when the buyer cannot physically access the product, e.g. through online shopping, the EPC of the products will be available from a website. As it will be discussed later in Section III-D.1, the current owner must provide a true EPC and his/her address because these information can be verified in the blockchain. Consumer C invokes getManufacturerAddress() with the EPC and obtains the manufacturer's address. Then, he/she invokes getProductStatus() and getCurrentOwner() to manufacturer M's PM. If the product status is Owned, and the obtained address is the current owner's address, then the buyer will decide to buy the product by paying its selling price.
2) The current owner issues a transaction shipProduct() with the EPC of the product to be transferred and the buyer's address.
3) When the product is received, the buyer issues a transaction receiveProduct() to the contract PM with interrogated EPCs.

OUTCOME

In this section, we first explain how counterfeits can be identified and avoided through the operational procedure of the proposed POMS. We then present and discuss various practical issues and cases regarding the actual operation of the POMS.

1) PROTOCOL VERIFICATION

Let us consider all the following possible situations where a party, which can be a new owner of a product, checks the genuineness of the product that he/she will receive from its current owner. Let us assume that this current owner is a counterfeiter and tries to sell/transfer counterfeits to the new owner. For simplicity, in what follows, we denote a current owner and a new owner as a seller and a buyer, respectively. In this case, three situations are possible:

1) The seller possesses counterfeits with fake EPCs;
2) The seller possesses counterfeits and knows their true EPCs but does not possess their ownership;
3) The seller owns the genuine product and its ownership and possesses a number of counterfeits too.

For the first situation, the buyer can refuse to buy a counterfeit since he/she can check the product information from its EPC before purchasing it. EPC itself includes product information, e.g. company prefix, item reference, and serial number and thus the party can verify whether or not the EPC is really associated with the product to be purchased. For the second one, although the buyer is convinced that the EPC is genuinely associated with the desired product, it is doubtful about whether or not the seller possesses the ownership of the product. The buyer first obtains the owner's address of this EPC by querying getCurrentOwner(). The seller will then claim that this address is his/her own. Hence, the buyer would like to confirm this claim by making the seller issue a transaction shipProduct(), since only the current owner can issue it. However, the seller might refuse to do so before the money is paid. In this case, there is another simple way for the buyer to check on this. After the buyer obtains the current owner's address, he/she generates a challenge message with a pseudo-random number generator and makes the seller sign it. If the signature is verified with the public key that generates the current owner's address, the buyer can be reasonably convinced that the seller is truly the owner of its address. However, in this situation where the counterfeiter is assumed not to possess the ownership of its EPC, the signature verification will surely fail. Hence the buyer can abort this deal. In the last situation, the buyer is certain that the seller possesses the ownership of the product. In this case, it is possible that a seller will ship one of its counterfeits and issue a transaction shipProduct() with its EPC. However, we argue that there is no economical merit for the counterfeiter to do so. Since the seller must transfer the ownership of its EPC to the buyer, the original genuine product and any other counterfeits with the same EPCs are no longer sold. Since, in general, counterfeits are much cheaper than the original products, the counterfeiter eventually makes a loss and thus there is no merit for the counterfeiter to do so.

2) MULTIPLE OWNERS CASE

In the previously described protocol validation subsection, it is implicitly assumed that only one party can claim the ownership of one product (EPC). However, it is possible that multiple parties co-possess one product so that the situation gets even more complex than the single owner case. Our POMS can deal with such case by storing multiple owners' addresses for each EPC in enrollProduct(). In addition, when the ownership of a product is transferred, a condition must be set beforehand, e.g. with agreements from all owners. This, of course, might be sometimes complex, as for example it is possible that the exact holding ratio is required.

In this case, such information must be also stored on the blockchain. Therefore, according to the ownership status, the data structure of owners must be altered.

3) ARBITRATION BETWEEN OWNER AND BUYER

Regarding the transactions, as is often the case with e-commerce, the following unfortunate situation might occur: A buyer pays actual money to the product’s owner while he/she does not ship the product and vice versa. In general, this problem is solved by introducing a trusted third party between a buyer and seller, namely escrow [31]. However, our POMS does not deal with escrow because it is out of scope in terms of ownership proof. Actually, if a fee is payed by a cryptocurrency, e.g. ETH, the escrow is realized without a trusted third party by specifying a clever contract rule so that a cheater loses his/her money.

4) OWNER’s PRIVACY

One may consider that since any owner address is stored in the PM, this could be a privacy issue for customers. However, this issue can be solved by the customers by generating new public/private key pairs for each product. Actually, because this process can be automated by the wallet application of Ethereum, the customers do not have to take any precautions.

5) IMPERSONATION AVOIDANCE

One may also consider the situation when an attacker illegally issues transactions by pretending to be a victim, e.g. issuing a transaction under the victim’s address. For example, let us consider the following case: Customer C1 has the ownership of product EPC1 and an attacker wants to steal the ownership of EPC1. In this case an attacker may illegally issue a transaction with shipProduct() by specifying its argument as EPC1. However, in Ethereum, each transaction must be signed with the sender’s private key that also generates his/her address. Therefore, unless a customer leaks his/her private key, nobody else can generate his/her valid signature. Clearly, impersonation can be avoided in this way.

6) CUSTOMER PARTICIPATION

We assume that every party involved with POMS follows the aforementioned protocol. Although it is possible that some parties, like ordinary customers, forget or even might be reluctant to issue shipProduct() and receiveProduct() transactions, it is fair to assume that all participating parties obey the protocol because of the financial benefit that POMS offers. As previously mentioned and will be emphasized in the next section, the main application of POMS is to be used for dealing with not very cheap products, e.g. branded goods. Hence, the ownership proof of such products should be very much desired by customers.

Conclusion and Future Scope

We have proposed a novel blockchain-based product ownership management system (POMS) for the post supply chain, which makes the efforts of counterfeiters to clone genuine tags redundant since they cannot prove the possession of products on this system. Firstly, the overall practical system requirements have been identified. Then, we have introduced a full-fledged protocol that enables each party, including supply chain partners and customers, to transfer and prove the ownership of RFID tag-attached products. An important advantage of the proposed POMS is that customers can reject the purchase of counterfeits, even with a genuine EPC, under the condition that the seller does not possess their ownership. The protocol validation has been shown the validity of our POMS. Based on the proposed protocol, a proof of concept experimental system has been implemented on the Ethereum platform.
There can be more improvements and extensions of the Project. The Future Scope of the project can be as follows,

1. Write an extensive test suite to test the utilities and functions of the contracts,
2. Audit the contracts for bugs and possible hacks by some professional cybersecurity or blockchain security firm,
3. Build a Dapp (Decentralized Application) to consume the contracts and provide a UI,
4. Incentivize the users to keep using the application to update the owners by introducing some tokenomics.

References

1. Investopedia : https://www.investopedia.com/terms/b/blockchain.asp
2. The Economist : https://www.economist.com/business/2015/07/30/counterfeitcom
3. Harvard Business Review : https://hbr.org/2019/05/how-luxury-brands-can-beat-counterfeiters
4. Ethereum : https://ethereum.org/en/