Fraud Buster: Tracking IRSF Using Blockchain While Protecting Business Confidentiality

Shuaicheng Ma
Emory University
sma30@emory.edu

Tamraparni Dasu
AT&T Labs - Research
tamr@research.att.com

Yaron Kanza
AT&T Labs - Research
kanza@research.att.com

Divesh Srivastava
AT&T Labs - Research
divesh@research.att.com

Li Xiong
Emory University
lxiong@emory.edu

ABSTRACT

Decentralized delivery of physical or digital items via a sequence of handover actions is common in telecommunication, supply chains, snail mail, email, etc. In decentralized delivery systems, items are passed between carriers, from source to destination, without a central control, and often, by carriers that belong to different organizations. Delivery failures could be due to faults or the result of malicious actions like fraud, e.g., in International Revenue Share Fraud (IRSF), international phone calls are dropped by fraudulent telecommunication carriers. Tracking item delivery can help detect faults and fraudulent behavior. But the sequence of carriers used for delivery of a specific item is often business confidential, and should be revealed only in case of fraud.

In this paper, we demonstrate a blockchain-based system, Fraud Buster, for confidential tracking of routes in a decentralized delivery system. In particular, we illustrate the ability to track handover of calls while preserving business confidentiality when detecting where calls were dropped. The paper makes the use of a permissioned blockchain for tracking the required information yet revealing only the necessary information, when a fraud occurs.

Keywords

IRSF, fraud detection, blockchain, business confidentiality

1. INTRODUCTION

In decentralized delivery systems, carriers that belong to different organizations deliver items to a specified destination, with no central control. Delivery could be of physical or digital items, e.g., letters, parcels, emails, IP packets, etc. Items are carried by a sequence of carriers from source to destination. Each delivery step is a handover of an item between a pair of carriers, and the delivery steps repeat until the item reaches the specified destination. International telephone calls are managed by such a delivery system, where the call route is established via a series of handover steps.

Example 1.1. *International phone calls go through several carriers. When a caller from the USA calls a number in Latvia, initially the service provider of the caller, say AT&T, would handle the call. A connection will be made with a company that can forward the call closer to the destination, say British Telecom (BT). BT will forward the call further, say to Orange Polska, which will deliver the call to Tet (Lattelecom) which will hand it over to Rigatta SIA, the carrier of the receiving number. Each carrier will make an independent business decision as to which carrier should be next in the chain, e.g., based on the requested handling fee. The customer who initiates the call pays its carrier, AT&T in this example. AT&T pays a fee to BT for handling the call, BT pays part of that to Orange Polska, which pays to Tet its share, and Tet pays to Rigatta SIA its fee for the connection with the call receiver.*

Handling international calls, as presented in Example 1.1, is a typical case of decentralized delivery. Other examples are (1) international mail services, where a chain of handoffs between companies is established for delivering letters and packages to the destination address, (2) email in which mail servers forward the messages, (3) computer networks in which IP packets are transferred between servers, and (4) various supply chains, where delivery of physical items happens through a series of handovers between carriers.

A delivery may fail, due to a fault in the delivery system or a malicious action. Aborting the delivery maliciously is typically part of a fraud or an attack. If that happens, it is useful to be able to discover where the delivery failure occurred, e.g., which carrier dropped the call, dropped the IP packet, or failed to deliver the item to the next hop in the chain. This requires recording delivery information in a trusted way, and coping with the following three challenges. First, the distribution system is decentralized, so there is no entity that has all the information about deliveries and carriers. Second, there might not be a single entity that all the involved organizations and individuals trust. Third, some of the information could be restricted by *business confidentiality*, as elaborated next.

Delivery chains in a decentralized system are often obscure or change frequently. For instance, the chain of carriers described in Example 1.1 could change if one of the carriers offered a cheaper price for handling the call. When BT needs

11th Annual Conference on Innovative Data Systems Research (CIDR '21) January 10-13, 2021, Chaminade, USA.

to choose which company would be the next in the sequence and handle the call, it may prefer a different company to Orange Polska, e.g., Teo LT of Lithuania, if the fee requested by Teo LT is lower than the fee of Orange Polska.

In decentralized delivery systems, the carriers are often reluctant to reveal information about the route and the handoffs. For instance, business confidentiality may prevent revelation of information about handoffs. In Example 1.1, BT may not want to reveal its selection of Orange Polska and the incurred fee, to negotiate a lower fee with Teo LT. In the case of IP routing, the route can often be discovered using `traceroute` (or `tracert`). However, if VPN is used some information about the route would remain concealed.

In this paper we demonstrate our Fraud Buster system for tracking deliveries in a decentralized delivery system by recording handoffs on a blockchain. Based on the records, the place where a fault occurred can be efficiently discovered while protecting business confidentiality. We illustrate the tradeoff between confidentiality and the ability to detect carriers involved in delivery failure. In particular, we demonstrate the use of our solution to mitigate IRSF—a fraud that costs billions of dollars to telecommunication companies.

The main contributions of the paper are as follows.

- Presenting the problem of confidentially tracking handovers in a decentralized delivery system, with IRSF as a particular use case.
- Illustrating the suitability of blockchain for the confidential tracking problem.
- Introducing four different confidentiality models and showing how to implement them on a blockchain.
- Demonstrating the effectiveness of the proposed solution on a decentralized system implemented using Mininet, which simulates a virtual SDN setting.

The paper is organized as follows. In Section 2, we describe IRSF and its effect on telecommunication companies. In Section 3, we present our framework, including definitions and notations, and we formally define the research problem. In Section 4, we discuss business confidentiality and present four different confidentiality models. The Fraud Buster system is presented in Section 5, and our demo is described in Section 6. Finally, in Section 7 we discuss our conclusions.

2. IRSF

International Revenue Share Fraud (IRSF) is the main motivating use case for our study. It is one of the most prevalent frauds plaguing the telecommunication industry [2, 10, 16]. A survey from 2017, conducted by the Communications Fraud Control Association[1] (CFCA), estimated that the revenue losses due to IRSF, for telecommunication companies worldwide, exceeded $10B yearly.[2]

IRSF occurs when a fraudulent international phone call is made and the fraudster, or an associate of the fraudster, is paid a portion of the cost of terminating the call. IRSF often entails an artificial inflation of traffic, i.e., traffic-pumping to international premium rate numbers (IPRN), or switching international calls to a fraudster carrier who drops the call, yet gets paid. The revenues of the premium number holder or of the carrier are shared with the fraudster. For instance, a PBX box can be hacked to issue many calls to the premium number, or via a carrier that participates in the fraud. In the first case, the fraudulent calls are often very long, because calls to a premium rate number are billed by duration of the call. In the second case, there is typically a high volume of very short calls. These are calls that are short stopped by the fraudster carrier. The fraudster carrier receives a small fee for each call and profits from "handling" many short calls that are dropped, i.e., not delivered to any end user.

When fraud is discovered, the records that are related to the fraud need to be identified. This is costly and today requires human labor, in particular, contacting other carriers to get the data related to the calls involved in the fraud. That data is stored in the form of Call Detail Records (CDRs). Since each carrier stores the CDRs of calls it handles in its own proprietary database, there is no single database that provides an overview of how calls were handled end-to-end, along the entire route from the initiator of the call to the end receiver. Furthermore, there is no mutually agreed upon trusted central entity that maintains such a comprehensive database across all carriers.

Example 2.1. *As an example of IRSF, consider a fraudster that hacks into a private telephone system (PBX) of a customer of telecom company T, and issues calls to premium numbers in Latveria*[3]*. Since the T customer has not made the call, company T adjusts the customer's bill by refunding the customer for this call. However, since T forwarded the call to other carriers, it has to pay them for their service. A fraction of the payment goes to the fraudster carrier who terminates (or claims to terminate) the calls. The customer whose PBX has been hacked is not paying for the fraudulent calls, but the carrier of this customer (company T) loses money. All the other involved carriers, and in particular the one who terminates the call or provides the premium number, make a profit. The fraudster carrier shares the revenues from such fraudulent calls with the hacker. Note that international calls to some countries are quite expensive.*

To reveal the fraudster carrier, suspicious calls are tracked, in order to discover the carriers that were part of the route, and if a call was dropped, find the carrier that short-stopped it. This requires access to information about the call. However, the information about each handover of the call is only stored in the databases of the involved carriers. Discovering the involved carriers, therefore, is an iterative and costly process. It would have been easier if there had been a central storage of all the information. But the different carriers are competitors and may not fully trust each other or let other carriers manage such information for them.

For coping with IRSF when carriers do not fully trust one another, we present a blockchain-based decentralized system that tracks dubious international calls, e.g., calls to suspicious numbers or a burst of calls to particular countries. The goal is to help track the termination of calls, and to mitigate fraud by making fraudulent behavior (or assistance to fraudulent behavior) traceable in an automatic way.

Example 2.2. *Consider the IRSF case described in Example 2.1. The first carrier, T, only knows the next carrier*

[1] https://www.cfca.org/

[2] https://gdpr.report/news/2017/05/29/telecommunications-battle-fraud/

[3] Fictional nation appearing in American comic books.

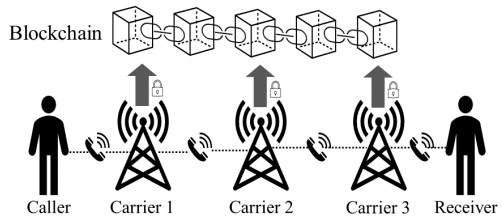

Figure 1: Handoff recordings on a blockchain

to which it forwarded the call, say BT. To discover the carrier to which BT transferred the call, company T needs to approach BT and ask for the information. BT would need to look for the relevant Call Detail Record (CDR) in its records and provide that. Suppose that from the CDR, company T learns that the next carrier in the chain is Orange Polska. This requires approaching Orange Polska and getting the CDR from them, to know who is the next carrier in the chain. This continues until the fraudster carrier is revealed or when a carrier refuses to cooperate. It is a slow and expensive process, which in some cases could cost more than the losses incurred by the fraud. If, however, all the information were already on a blockchain protected from tampering attempts of the fraudster, it would be easier to extract it and identify the carriers that handled the fraudulent calls.

In this paper we consider four undesirable behaviors of carriers: (1) short stopping a call and not recording the call on the blockchain, (2) short stopping the call and adding a fake record to the blockchain as if a successful handoff has been executed, (3) handling the call properly but adding a false record to the blockchain, (4) handling the call properly without recording that on the blockchain. Our goal is to use a blockchain for revealing the shortstops while ensuring confidentiality in non-fraudulent cases.

3. FRAMEWORK

We now present our framework, including terminology, notations and problem definition.

Decentralized delivery system. A *decentralized delivery system* consists of a network of *carriers* and dispatched items. The items can be physical (e.g., parcel) or virtual (e.g., email, international call). Each dispatch (call) is between a pair of end users (sender-receiver/caller-callee).

Let C be the set of carriers and L be the set of links between the carriers. Let U be a set of *end users*, which could be senders (callers) and receivers (callees). A link between carriers c_1 and c_2 enables handover of items from carrier c_1 to carrier c_2. The set of carriers C and links L yields a directed graph $G = (C, L)$ where C are the nodes and L are the edges. Each end user can also handover an item to a carrier or receive an item from a carrier, but end users do not serve as intermediary carriers.

A *delivery task* is the duty to deliver an item or establish a phone call connection from end user $u_s \in U$ (sender) to end user $u_r \in U$ (receiver). This is executed using a set of carriers, which could belong to different organizations. A successful delivery is a sequence of handoffs that creates a path in G from a carrier connected to u_s to a carrier connected to u_r. Different methods can be used for the link selection per delivery task, e.g., based on estimations of the shortest path to the destination, the load on carriers or the handling fee. For instance, when forwarding a call, BT may select between Orange France, Orange Polska and Deutsche Telekom based on the fee each company charges for handling the call to the desired destination.

Successful delivery is a sequence $u_s, c_1, c_2, \ldots, c_n, u_r$ where

- end users $u_s \in U$ and $u_r \in U$ are sender and receiver;
- each pair of consecutive carriers c_i and c_{i+1} are linked, i.e., $(c_i, c_{i+1}) \in L$, and there is a successful handoff between c_i and c_{i+1};
- the u_s to c_1 and c_n to u_r handoffs are successful.

A *failed delivery* from sender u_s to receiver u_r is a sequence of handoffs $u_s, c_1, c_2, \ldots, c_i$ that starts in u_s but does not reach u_r. We refer to c_1 and c_i as the *first carrier* and *last carrier* of the failed delivery, respectively.

In the case of IRSF, only the first carrier loses money because it needs to compensate the sender and pay c_2 for the service. (Note that c_2 also needs to pay c_3 for the service, but it still makes a small profit, and the same is true for the other carriers c_j, $2 \leq j < i$.) Carrier i, the fraudster, receives a fee for handling the call but drops it and does not pay carrier c_{i+1} because there is no handoff of the call. The fraudster shares the revenue with the hacker that initiated the call. Carrier i could also be a carrier who makes the connection to a fraudster IPRN.

EXAMPLE 3.1. *Consider a successful delivery through four carriers $u_s, c_1, c_2, c_3, c_4, u_r$. User u_s pays \$1.8 to the home carrier c_1, for the call. Carrier c_1 pays \$1.2 to carrier c_2 for handling the call and gains \$0.6, carrier c_2 pays \$0.8 to c_3 and profits \$0.4, carrier c_3 pays \$0.4 to c_4 and keeps \$0.4, and c_4 delivers the call to u_r.*

Now, consider a case where c_3 short stopped the call. The sequence is u_s, c_1, c_2, c_3. The carrier c_1 does not charge u_s for the call because it is not a genuine call of u_s. Carrier c_1 still pays \$1.2 to c_2 for handling the call, and c_2 pays \$0.8 to c_3. Since c_3 dropped the call, it does not need to pay to any other carrier. In this case, c_1 loses \$1.2, c_2 gets \$0.4 and c_3 gains \$0.8. A burst of 1000 such calls would lead to the case where c_1 loses \$1200 and c_3 gains \$800. A portion of the revenue of c_3 is shared with the hacker who hacked the PBX phone system and initiated the calls.

Registry using blockchain. To mitigate IRSF, the last carrier of suspected fraud calls (failed delivery) should be identified and made known to the first carrier, e.g., in the case of IRSF it could spare paying the fraudster carrier, and in the case of a technical failure, discover the carrier responsible for it.

A registry system that records all the handoffs could help in detecting the failure point. But managing such a system is challenging given that the delivery system comprises of carriers that may not fully trust each other. The registry should be trustworthy and should not be controlled by any single carrier. In a registry system that is controlled by a single organization, the controlling organization can deny access to the information from other parties. In a registry system that has replicas stored on different nodes independently, it is difficult to guarantee that exactly the same information

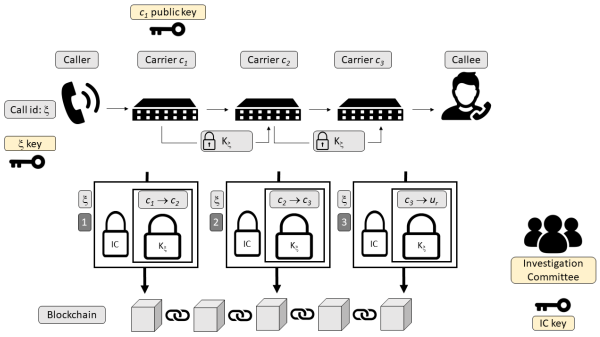

Figure 2: Recording handoffs under the All-to-All confidentiality model, where the handoffs are encrypted using a unique key created for each call, and the key is securely transferred to the carrier handling the call. The handoffs are encrypted twice—first with the unique key of the call and then with the key of the IC. The Last-to-All model is similar except that if the call is dropped, the IC only decrypts the last two records.

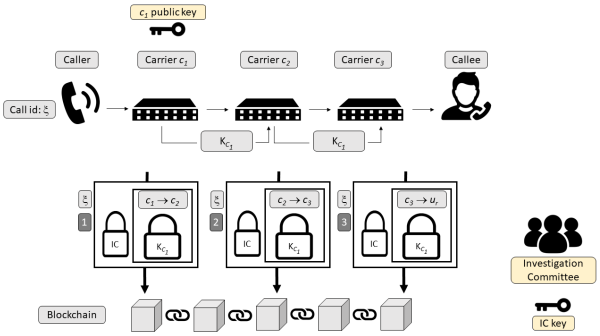

Figure 3: Recording handoffs under the All-to-First confidentiality model, where the handoffs are encrypted twice—first with the key of the first carrier and then with the key of the IC. The Last-to-First model is imlemented in the same way, except that if the call is dropped, the IC only decrypts the last two records.

appears in all replicas. Therefore, we propose a solution based on blockchain.

Blockchain is a decentralized, tamper-proof and transparent ledger, managed by peers that are part of a peer-to-peer network [12]. The peers create blocks of transactions and add them to the chain in a way that guarantees consensus regarding valid transactions and their order. Blockchain was initially developed to prevent the "double spending" problem in cryptocurrencies [12], but recently, there has been a growing interest in using blockchains for a variety of applications where there is a need to reach consensus in a decentralized environment [1, 5, 11, 18, 19]. In the case of a permissioned blockchain, the set of peers is known and a consensus mechanism is used to decide which blocks are valid and can be added to the blockchain [7].

In our system, a blockchain is used as a ledger to record delivery tasks and handoffs, see Fig. 1. It provides a tamper-proof decentralized log of the delivery tasks and their execution, and these records can be used to track handoffs and discover points of failure, i.e., the carrier that dropped the call (or failed to handover an item). In this paper we assume that the blockchain peers are entities that were assigned to manage the blockchain. They can be carriers, but do not have to be carriers or end users.

Confidentiality. Blockchains provide transparency, where all the peers that manage the blockchain see the stored transactions. This is an advantage in many applications. However, there is a conflict between transparency and business confidentiality. Carriers may not want information about their handoffs to be revealed to their competitors.

To achieve business confidentially, the following two general guidelines should be applied:

1. *minimize the exposed information*, and
2. *minimize the number of access permissions.*

Information should only be revealed when necessary and only to entities that need to see the information.

When minimizing the exposed information, the goal is to reduce the number of handoffs that are revealed to a third party. An example of such a restriction is to only reveal transactions that are part of a failed delivery. When minimizing the number of viewers, the set of carriers (or other entities) that are privy to the disclosed information should be as small as possible. For example, if a carrier does not participate in a delivery, it should not be exposed to information about that delivery.

In this paper we consider four confidentiality models, for a failed delivery $u_s, c_1, c_2, \ldots, c_i$.

- **All-to-All:** The handoffs of the delivery are revealed to the carriers $c_1, c_2, \ldots, c_i$.
- **All-to-First:** The handoffs of the delivery are revealed to the first carrier c_1.
- **Last-to-All:** Only the handoffs associated with the last two carriers c_{i-1} and c_i are revealed to the carriers $c_1, c_2, \ldots, c_i$.
- **Last-to-First:** Only the handoffs associated with the last two carriers c_{i-1} and c_i are revealed and only to the first carrier c_1.

Disclosing the handoffs to all the carriers on the path allows all of them to know that they are part of a failed delivery, so that they collectively could be responsible for the prevention of future failures. Revealing the information just to the first carrier provides stronger confidentiality. Similarly, revealing the entire set of handoffs on a route provides information that could be used to prevent reoccurring failures. Revealing just the last two carriers limits the exposure to only a small set of carriers and provides stronger confidentiality. Note that in a case of a malicious action, the carrier that drops a call or fails to deliver an item, say c_{i-1}, may try to conceal that by recording a handoff to c_i on the blockchain. In this case, information about both c_{i-1} and c_i should be revealed, to examine which one is responsible for the short stop. We consider four behaviors of the carriers.

- **Honest** (but curious): the carrier transfers the call and correctly records that on the blockchain.

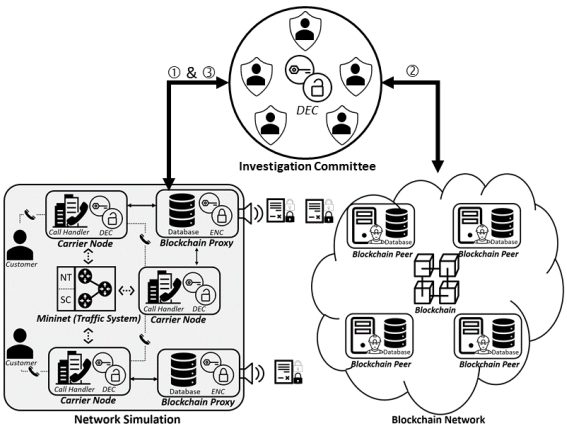

Figure 4: System architecture

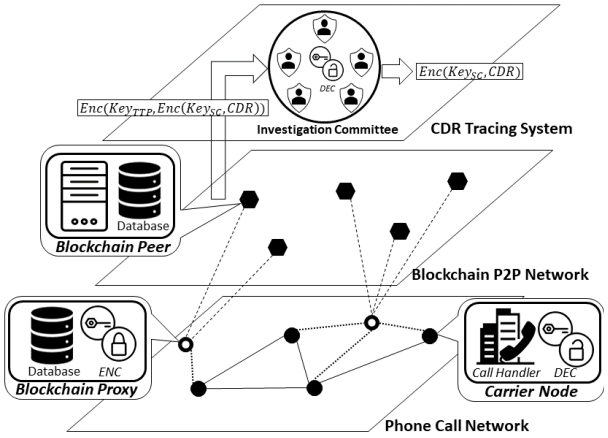

Figure 5: Layered network

- **Fraudulent**: the carrier drops the call but records a fake transfer on the blockchain.
- **Sloppy**: the carrier transfers the call but fails to properly record that on the blockchain.
- **Malicious**: the carrier drops the call and records nothing on the blockchain.

A fraudulent carrier c_{i-1} may add to the blockchain a fake record that the call was forwarded to c_i. This is the reason for revealing the last two nodes of a failed delivery. Typically, the majority of the carriers are honest, but even the honest carriers should not be privy to confidential information (handoffs of other carriers).

Goal. Demonstrate a system that implements the four confidentiality models on top of a blockchain, in the presence of all four carrier behaviors.

4. CONFIDENTIALITY MODELS

In this section, we discuss the implementation of the confidentiality models in FRAUD BUSTER. In all the models, information about handoffs is initially concealed. We assume that there is a committee that is responsible for deciding when information should be revealed, and we refer to it as the *investigation committee* (IC). The IC consists of several independent entities. It is honest but curious. That is, the IC can be trusted to decide when information should be revealed but should not see any confidential information. It can be a proxy that executes a court order, or can be some other group of semi-trusted entities. It can be implemented as a single node, as a distributed system with a consensus protocol, by the blockchain peers, etc.

The protocols use the RSA [15] public key cryptosystem [9], and each carrier c has a pair (K^c_{priv}, K^c_{pub}) of private and public keys. The public key can be used for encrypting of short texts, e.g., using optimal asymmetric encryption padding [3]. Given a message m, we denote by $\text{Enc}(m; K)$ and $\text{Dec}(m; K)$ the encryption and decryption of m using key K. Given a message m, a carrier c can sign m by applying to m a cryptographic hash function $h()$ like SHA2 [14], and encrypting the result using its private key, $s = \text{Sign}(m) = \text{Enc}(h(m); K^c_{priv})$. Any carrier or blockchain peer could validate a signature by checking that $h(m) = \text{Dec}(s; K^c_{pub})$. The IC $\mathcal{C}$ also has a pair $K^{\mathcal{C}}_{priv}$, $K^{\mathcal{C}}_{pub}$ of private and public keys.

Encrypting a long plaintext using a public key is inefficient. So, often for a message m and a public key K_{pub}, the encryption is a two step process using a symmetric key like AES [4]. The symmetric key is used for both the encryption and the decryption [13]. First, a new secret symmetric key K_{sym} is created. The symmetric key is used to encrypt the message and the public key is used to encrypt the symmetric key. The two records $\text{Enc}(m; K_{sym})$ and $\text{Enc}(K_{sym}; K_{pub})$ are added to the blockchain. We denote this 2-step encryption by $\text{Enc2}(m, K_{pub})$.

All-to-All. Given a call with call id ξ, from sender u_s to receiver u_r, the route is recorded by storing handoffs on the blockchain. The handoffs are encrypted by a key that is shared with the relevant carriers, to prevent exposing the information to carriers that are not part of the route of call ξ. The encryption is by using a symmetric key K^{ξ}_{sym}, uniquely created for call ξ, e.g., using AES [4].

Carrier c_1 creates the key K^{ξ}_{sym}. In step i, carrier c_i records on the blockchain the encrypted handoff information $(\xi, \text{Enc2}(r; K^{\mathcal{C}}_{pub}))$, where $r = \text{Enc}((c_i, c_{i+1}, \text{CDR}^i_{\xi}); K^{\xi}_{sym})$ and CDR^i_{ξ} is the call detail record of call ξ when handled by carrier c_i. The CDR contains details about the call such as time, duration, completion status, source and destination numbers, etc. Carrier c_i hands over the encrypted key K^{ξ}_{sym}, to c_{i+1}. That is, c_i encrypts the key using the public key of c_{i+1} and sends the encrypted key $\text{Enc}(K^{\xi}_{sym}; K^{c_{i+1}}_{pub})$ to c_{i+1}. Carrier c_{i+1} can decrypt the message using its private key to discover K^{ξ}_{sym}. Note that the handoff information is encrypted twice, first using the key created by the first carrier and then by the public key of the IC. By the end of this process, the carriers involved in handling the call ξ store the encrypted handoffs on the blockchain. An example of a call and the records that are stored on the blockchain are depicted in Fig. 2.

When the handoffs of call ξ need to be disclosed due to an indication that ξ was short stopped, the IC $\mathcal{C}$ decrypts all the transactions with call identifier ξ and adds the decrypted transactions to the blockchain. The decrypted transactions are still encrypted by K^{ξ}_{sym}, so $\mathcal{C}$ and all the carriers that are not part of call ξ cannot see the handoffs. Carriers that

Figure 6: Mininet control: the simulator creates the network nodes, the links, i.e., the connections between nodes, and the routing tables for the nodes. In the simulation, calls are simulated by IP packets, which are routed according to the routing tables. Fraudulent or malicious nodes may drop the packets, as an execution of a short stop.

have K^{ξ}_{sym} can decrypt the handoff records of ξ.

There is no single carrier that stores all the information, can change stored transactions or can deny access to stored transactions. Moreover, the IC does not need to store or see any information on calls.

All-to-First. To change All-to-All into All-to-First, the public key $K^{c_1}_{pub}$ of the first carrier c_1 is used instead of the key K^{ξ}_{sym}. In the handoffs, each carrier c_i instructs the next carrier c_{i+1} to use $K^{c_1}_{pub}$. The record stored by c_i on the blockchain has the form

$$(\xi, \text{Enc2}(\text{Enc2}((c_i, c_{i+1}, \text{CDR}^i_{\xi}); K^{c_1}_{pub}); K^{\mathcal{C}}_{pub})).$$

An example of a call and the records that are stored on the blockchain are depicted in Fig. 3. Only c_1 has the private key to decrypt the transactions, so only c_1 can see the handoffs. However, since transactions are also encrypted using the key of the IC, carrier c_1 can see the handoffs only after the decryption of the information by IC.

Last-to-All. In this model, the last two handoffs of ξ should be identified. This is done by adding a handoff number to the records, where carrier c_i records on the blockchain

$$\hat{\xi}_i = (\xi, i, \text{Enc2}(\text{Enc}((c_i, c_{i+1}, \text{CDR}^i_{\xi}); K^{\xi}_{sym}); K^{\mathcal{C}}_{pub})).$$

As integrity constraint, the blockchain peers verify when adding a record (ξ, i, m) to the blockchain that it contains a record of the form $(\xi, i-1, m')$ and does not contain a record of the form (ξ, i, m''), for any cipher texts (encrypted content) m, m' and m''. The transfer of the shared key K^{ξ}_{sym} is the same as the key transfer in All-to-All.

When a call ξ seems to be part of a fraud (the sender u_s claims it is the result of a hack and the first carrier c_1 needs to investigate it), the IC $\mathcal{C}$ decrypts the last two records of ξ and sends to c_1 the records $\text{Dec}(\hat{\xi_{i-1}}; K^{\mathcal{C}}_{priv})$ and $\text{Dec}(\hat{\xi_i}; K^{\mathcal{C}}_{priv})$. The carrier c_1 would still need to investigate

Carrier2

Cid	S->D
h2_1	2->10
h2_2	2->3
h2_3	2->7
h2_4	2->8
h2_5	2->10

Proxy2

Cid	S->R	P-C-N
h2_1	2->10	s-2-1
h4_1	4->1	4-2-1
h6_1	6->2	1-2-r

Carrier3

Cid	S->D
h3_1	3->1
h3_2	3->1
h3_3	3->7
h3_4	3->1
h3_5	3->10

Proxy3

Cid	S->R	P-C-N

Carrier4

Cid	S->D
h4_1	4->1
h4_2	4->1
h4_3	4->5

Proxy4

Cid	S->R	P-C-N
h4_1	4->1	s-4-2

Carrier5

Cid	S->D
h5_1	5->10
h5_2	5->2
h5_3	5->9
h5_4	5->8
h5_5	5->2

Proxy5

Cid	S->R	P-C-N
h5_1	5->10	s-5-1
h8_1	8->5	1-5-r

Figure 7: Handover transactions that four carriers and their proxies see. In real time, carriers and proxies only see handovers that they are part of.

Carrier2

Cid	S->D
h2_15	2->5
h2_16	2->5
h2_17	2->9
h2_18	2->7
h2_19	2->10
h2_20	2->7

Proxy2

Cid	S->R	P-C-N
h3_14	3->4	1-2-4
h9_14	9->2	4-2-r
h10_1510	->2	1-2-r
h2_15	2->5	s-2-1
h5_16	5->4	1-2-4
h2_16	2->5	s-2-1

Carrier3

Cid	S->D
h3_14	3->4
h3_15	3->1
h3_16	3->4
h3_17	3->6
h3_18	3->6
h3_19	3->2

Proxy3

Cid	S->R	P-C-N
h7_12	7->3	5-3-r
h1_13	1->3	5-3-r
h3_13	3->2	s-3-5
h3_14	3->4	s-3-5
h1_15	1->3	5-3-r
h3_15	3->1	s-3-5

Carrier4

Cid	S->D
h4_14	4->10
h4_15	4->7
h4_16	4->3
h4_17	4->8
h4_18	4->5
h4_19	4->3
h4_20	4->6

Proxy4

Cid	S->R	P-C-N
h3_14	3->4	2-4-r
h9_14	9->2	9-4-2
h8_15	8->4	9-4-r
h4_15	4->7	s-4-9
h9_15	9->2	9-4-2
h10_1610	->4	9-4-r
h5_16	5->4	2-4-r

Carrier5

Cid	S->D
h5_14	5->4
h5_15	5->8
h5_16	5->4
h5_17	5->4
h5_18	5->2
h5_19	5->4
h5_20	5->10

Proxy5

Cid	S->R	P-C-N
h3_14	3->4	3-5-1
h5_15	5->8	s-5-1
h2_15	2->5	1-5-r
h1_15	1->3	1-5-3
h3_15	3->1	3-5-1
h5_16	5->4	s-5-1
h2_16	2->5	1-5-r

Figure 8: The setting of Fig. 7 after a while. The handover transactions of call h3_14 are circled, to show the route of this call.

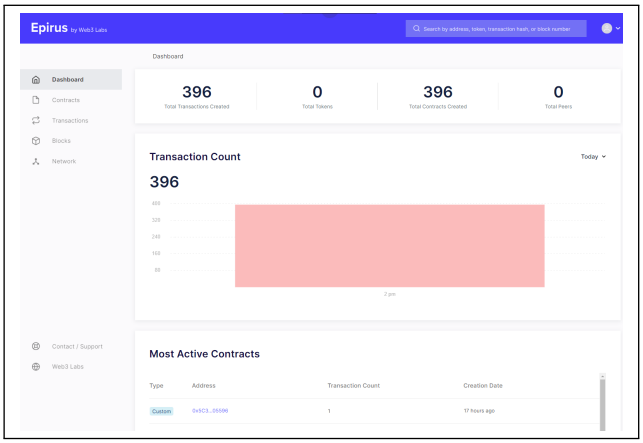

Figure 9: Blockchain dashboard.

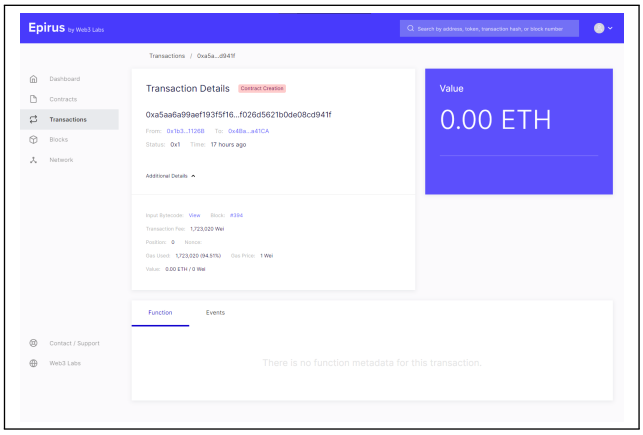

Figure 10: Transaction details.

which one of the two carriers c_i and c_{i+1} is the fraudster. It could be that c_i is a fraudster that added a fake transfer record ξ_i, but in this case the record ξ_{i-1} could provide the information about the handoff of the call to c_i. Another case is that c_{i+1} is malicious and it will not add any record to the blockchain. In this case, the record ξ_i will provide information about the handoff of the call to c_{i+1}. Carrier c_1 cannot distinguish between these cases without an investigation. In all these cases, only records decrypted by $\mathcal{C}$ are visible and only to the carriers that hold the key K^{ξ}_{sym}.

Last-to-First. Implementing the Last-to-First model is the same as the implementation of the Last-to-All model except that the public key of the first carrier $K^{c_1}_{pub}$ is used instead of the shared key of the call K^{ξ}_{sym}.

Using Secret Sharing for the IC. When the IC consists of several entities, deciding when to decrypt records should be based on a consensus mechanism, e.g., for an IC of size n and some $0 < k \leq n$, requiring that any subset of k members of the committee could decrypt records on the blockchain, but a smaller subset would not be able to do so. This can be implemented using *secret sharing*. In FRAUD BUSTER, we are using Shamir's secret sharing [17].

In Shamir's secret sharing, first a large prime number p is selected and all the computations are executed modulo p, that is, with respect to the finite field $\mathbb{Z}_p = \mathbb{Z}/p\mathbb{Z}$. The secret is a number $s \in \mathbb{Z}_p$. The secret holder creates $k-1$ random numbers $a_1, \ldots a_{k-1}$, where $a_i \in \mathbb{Z}_p$ for $1 \leq i < k$. These numbers define a polynomial $f(x) = s + \sum_{i=1}^{k-1} a_i x^i$ of degree $k-1$. Then it sends to each peer j the encrypted secret point $\text{Enc}((j, f(j)); K^{peer_j}_{pub})$. Any set of k or more peers can reveal their points to each other, and using interpolation the polynomial $f(x)$ can be computed, to discover the secret s. Any group of less than k peers cannot discover the secret s even if they reveal to each other the points they received.

The goal is to create and disseminate a shared secret for the committee members in $\mathcal{C}$. This is done as follows. Consider the delivery $u_s, c_1, c_2, \ldots, c_i, \ldots$ of call ξ, where in each handover, the carrier records encrypted CDR and delivery details, on the blockchain. When carrier c_i encrypts the information, it uses the following protocol.

1. Carrier c_i generates a new pair of matching public and private keys $(K^{c_i,\xi}_{pub}, K^{c_i,\xi}_{priv})$.

2. Carrier c_i generates a secret sharing scheme where the secret is $s = K^{c_i,\xi}_{priv}$. It generates a polynomial $f_i(x)$ of degree k over $\mathbb{Z}_p$ where $f_i(0) = s$, and sends to each peer j the encrypted pair $\text{Enc}((j, f(j)); K^{peer_j}_{pub})$.

3. Carrier c_i adds to the blockchain the record

$$(\xi, i, \text{Enc2}(\text{Enc2}((c_i, c_{i+1}, \text{CDR}^i_{\xi}); K^{c_1}_{pub}); K^{c_i,\xi}_{pub}))$$

under the models All-to-First and Last-to-First; and

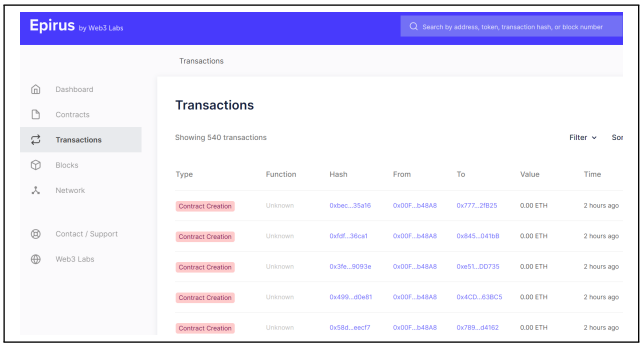

Figure 11: Transaction explorer.

it adds to the blockchain the record

$$(\xi, i, \text{Enc2}(\text{Enc}((c_i, c_{i+1}, \text{CDR}^i_\xi); K^\xi_{sym}); K^{c_i,\xi}_{pub}))$$

under the models All-to-All and Last-to-All.

Decryption requires k or more members of the IC, to reveal their share of the secret, compute the decryption key $K^{c_i,\xi}_{priv}$ and use it to decrypt the record on the blockchain. The decrypted records are still encrypted by key $K^{c_1}_{pub}$ in the All-to-First and Last-to-First models, or by K^ξ_{sym} in the All-to-All and Last-to-All models. In the first case, only the carrier c_1 can decrypt them using its private key. In the second case, all the carriers on the route, who received K^ξ_{sym}, can decrypt the information. The information is not revealed to any other entities, including members of the IC.

5. FRAUD BUSTER SYSTEM

We now describe the FRAUD BUSTER system. The system architecture is depicted in Fig. 4. It has three parts: (1) a network simulator, (2) the blockchain managed by the blockchain peers, and (3) the investigation committee (IC) component. This can be seen as a layered network where the lowermost layer consists of the network and the transfer of calls, the middle layer is the recording of call handoffs on the blockchain. The top layer is the application of tracking call routes confidentially. The layers are depicted in Fig. 5.

Network simulator. Network simulation is implemented using Mininet (http://mininet.org/), a simulator of virtual networks with Software Defined Networking (SDN) capabilities. The SDN capabilities provide control over the network traffic at the packet level, using OpenFlow commands executed on Open vSwitches (https://www.openvswitch.org/). Dispatched IP packets simulate calls. The Open vSwitches represent carriers and different behaviors (honest, fraudulent, sloppy, malicious) are assigned to them. Ryu (https://ryu.readthedocs.io/) is used for the SDN controller, to control the traffic flow according to the provided specifications. See illustration of the Mininet control screen in Fig. 6. A Network Topology Specification and Configuration (NTSC) was implemented, to control different network parameters, including the network topology. For scalable storage and exploration of the network, a graph database may be used [8].

Each carrier is connected to a blockchain proxy and has an encryption/decryption module. The blockchain proxy manages the connection of the carrier with the blockchain peers and supports access to the blockchain. To facilitate data extraction and evaluation of queries over the information in the blockchain, a PostgreSQL RDBMS is used by each proxy, to manage local data. Note that the blockchain proxies only deliver encrypted data, so the same proxy can serve more than one carrier.

Blockchain and P2P network. The system uses a permissioned version of Ethereum (https://ethereum.org/) with a Proof-of-Authority (PoA) consensus protocol. However, any other blockchain system or any other consensus protocol can be used for the decentralized trusted storage of handoffs. In PoA, a small set of trusted validators create the blocks. In each round, a leader is selected randomly from the set of validators. The leader suggests a block. If the block is approved by the majority of the validators, it is added to the blockchain. Otherwise, the leader is considered malicious and removed from the set of validators. PoA has a high throughput (transaction rate) in comparison to common consensus protocols like proof of work (PoW), however, it is considered less secure than PoW. Improving scalability, e.g., by chain partitioning as suggested in [6], is an ongoing research direction.

Figures 7 and 8 depict the handoff transactions that four carriers see. As time passes, more handoffs are conducted and the carriers see more transactions. To illustrate that, we present in Figures 7 and 8 the transactions at two different times. The table of each carrier contains the columns `Cid` and `S->D`, where `Cid` is the call identifier and `S->D` is the pair of source and destination carriers of the call. For example, in the first row of the table of Carrier 2 in Fig. 7, `h2_1` is the call id, and `2->10` specifies that the source and destination of the call are carriers 2 and 10. The Proxy tables show transactions that go via the proxy of the Carrier. In the proxy table there are three columns. The `Cid` column is the call identifier, the `S->R` column specifies the sender and receiver nodes of the call, and `P-C-N` refers to the previous, current and next nodes in the path to the destination. The current node appears in orange. For example, the first row of Proxy 2 in Fig. 7 contains the call id `h2_1`, the source and destination pair 2 and 10, as `2->10`, and the 3-tuple `s-2-1`, which specifies that the previous node is the sender, the current node is Carrier 2 and the next node is Carrier 1. Note that having the value '`s`' for the previous node refers to the sender user (the caller), and the value '`r`' as the next node refers to the receiver user (the callee).

In Fig. 8, the transactions of call `h3_14` are circles. This call is routed from Carrier 3 to Carrier 4, based on the value `3->4` in column `S->R`. Initially, in Proxy 3, we see `s-3-5` for `P-C-N`, which means that from Carrier 3, the call is forwarded to Carrier 5. In Proxy 5 we see `3-5-1`, which means that the previous node was Carrier 3, the current node is Carrier 5 and the next node is Carrier 1. Note that Proxy 3 and Carrier 3 do not see the forwarding of the call from Carrier 5 to Carrier 1. The records `1-2-4` in Proxy 2 and `2-4-r` in Proxy 4 show that the call was forwarded from Carrier 1 to Carrier 4 and from Carrier 4 to the receiver user. All the records together reveal the route `s->3->5->1->2->4->r` of the call, however, each carrier and proxy only see a local view of the handovers they where involved in. In other words, each carrier has a different view of the information on the blockchain because a carrier can only see its own records. Only when the records of a short stopped call are decrypted by the IC, the first carrier of the call can view them.

A blockchain dashboard allows tracking the information

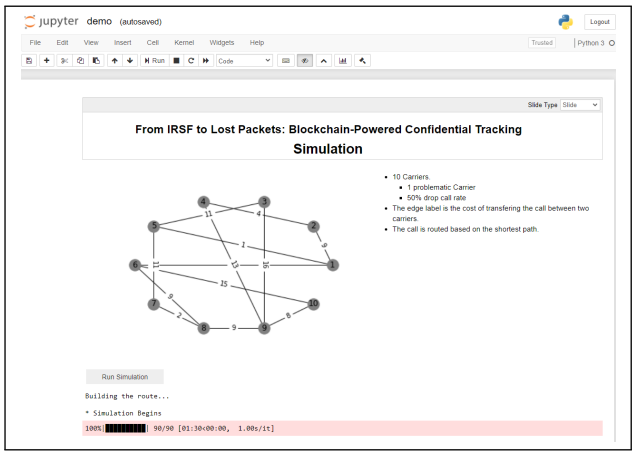

Figure 12: Step 1 of the simulation: the network is created, by defining nodes and links, and node behaviors (e.g., fraudster or malicious) are selected.

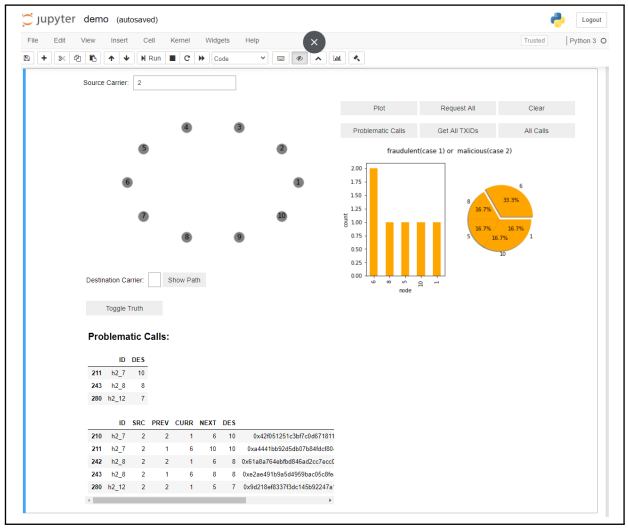

Figure 13: Step 2: calls are dispatched and the handover transactions are recorded on the blockchain. Carriers only see handovers they were part of.

on the blockchain, including statistics (Fig. 9), detailed information on selected transactions (Fig. 10) and the list of transactions in the blockchain (Fig. 11).

IC. The IC and the four confidentiality models are implemented in Python. The packages Cryptography and PyCryptodome are used for the cryptographic functions. When CDRs of a call ξ should be revealed, (1) the IC receives the call id from the proxy, (2) it retrieves the records of ξ from the blockchain and decrypts the relevant records, according to the confidentiality model, and (3) the decrypted records are shared with the relevant carriers through the proxy.

6. DEMONSTRATION

The demonstration will focus on showing two things—usability and confidentiality. Usability requires that in the case of a failed delivery, the records on the blockchain and the IC decryption are sufficient for revealing the place where the fraud occurred (indicating the last two nodes in the route). The demo will illustrate the usability for different behaviors of carriers (honest, fraudster, sloppy and malicious), under the assumption that most of the carriers are honest, like in the real world.

To illustrate confidentiality, the information that each carrier sees will be presented. We show that the view of each carrier does not lead to a confidentiality breach and carriers see in an unencrypted form only information they are allowed to see according to the model.

Figures 12–15 present Jupyter Notebook screenshots that depict a demonstrated scenario. These screenshots are updated in real time while the system runs and they provide information about the status of different components. Initially, the network is created based on given parameters, and the behavior of each node (honest, fraudster, sloppy or malicious) is selected. Fig. 12 presents a network topology that is created in Step 1, and a case where one of the carriers drops 50% of the calls through it. Initially, the identity and type of behavior of this carrier are not known to the other carriers. The simulation dispatches calls and routes them to the specified destination over the simulated network. Fig. 13 and Fig. 14 show the information that Carrier 2 sees at two times. This is a limited view of the handovers due to confidentiality. The carrier only sees handovers it is part of. The encrypted handoffs of all the honest carriers are recorded on the blockchain. Fig. 15 presents the decryption process and the revealed path of a dropped call.

7. CONCLUSION

This demonstration focuses on a specific but large problem, mitigation of IRSF—a problem at the scale of billions of dollars. However, the ability to track delivery failures and frauds in decentralized delivery systems has many additional applications in different supply chains and delivery systems. Our main contribution is showing that tracking can be done in a decentralized system, with limited trust between organizations, while maintaining business confidentiality. Future work includes adding economic intensives for carriers to record information on the blockchain, and incentives not to misuse the system, e.g., by requiring that carriers would pay for information.

Acknowledgment

We thank Teddy Chu from AT&T for his advice during this project and for providing us with information on IRSF and fraud investigation.

8. REFERENCES

[1] Arshdeep Bahga and Vijay Madisetti. *Blockchain applications: a hands-on approach*. VPT, 2017.

[2] Dan Baker. International revenue share fraud: are we winning the battle against telecom pirates? *Black Swan Telecom Journal*, 2012.

[3] Mihir Bellare and Phillip Rogaway. Optimal asymmetric encryption. In *Workshop on the Theory and Application of of Cryptographic Techniques*, pages 92–111. Springer, 1994.

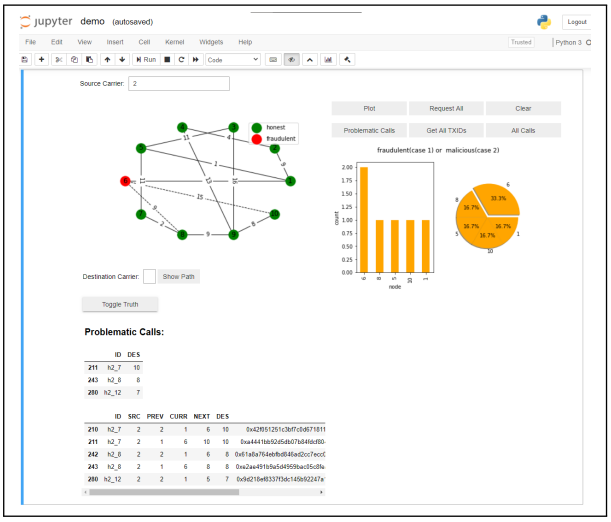

Figure 14: Step 3: information on calls that are short-stopped is collected, but it is still unknown to the first carrier where calls were dropped.

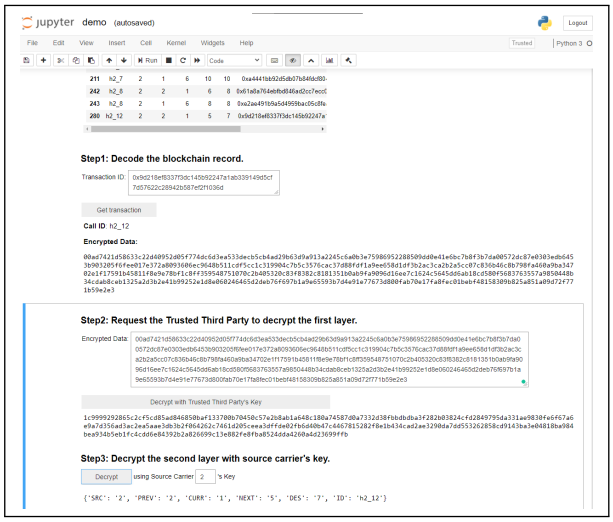

Figure 15: Step 4: the blockchain records of a dropped call are decrypted and revealed to the first carrier of that call.

[4] Joan Daemen and Vincent Rijmen. *The design of Rijndael: AES-the advanced encryption standard*. Springer Science & Business Media, 2013.

[5] Tamraparni Dasu, Yaron Kanza, and Divesh Srivastava. Geofences in the sky: herding drones with blockchains and 5G. In *Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, 2018.

[6] Tamraparni Dasu, Yaron Kanza, and Divesh Srivastava. Unchain your blockchain. In *Proc. Symposium on Foundations and Applications of Blockchain*, pages 16–23, 2018.

[7] Tien Tuan Anh Dinh, Ji Wang, Gang Chen, Rui Liu, Beng Chin Ooi, and Kian-Lee Tan. BLOCKBENCH: a framework for analyzing private blockchains. In *Proceedings of the 2017 ACM International Conference on Management of Data*, SIGMOD '17. ACM, 2017.

[8] Pramod Jamkhedkar, Theodore Johnson, Yaron Kanza, Aman Shaikh, N. K. Shankaranarayanan, and Vladislav Shkapenyuk. A graph database for a virtualized network infrastructure. In *Proceedings of the 2018 ACM International Conference on Management of Data*, SIGMOD '18. ACM, 2018.

[9] Neal Koblitz and Alfred J Menezes. A survey of public-key cryptosystems. *SIAM review*, 46(4):599–634, 2004.

[10] Godfred Yaw Koi-Akrofi, Joyce Koi-Akrofi, Daniel Adjei Odai, and Eric Okyere Twum. Global telecommunications fraud trend analysis. *International Journal of Innovation and Applied Studies*, 25(3):940–947, 2019.

[11] Thomas McGhin, Kim-Kwang Raymond Choo, Charles Zhechao Liu, and Debiao He. Blockchain in healthcare applications: Research challenges and opportunities. *Journal of Network and Computer Applications*, 135:62–75, 2019.

[12] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. Technical report, Manubot, 2019.

[13] Wouter Penard and Tim van Werkhoven. On the secure hash algorithm family. *Cryptography in Context*, pages 1–18, 2008.

[14] Dian Rachmawati, JT Tarigan, and ABC Ginting. A comparative study of Message Digest 5 (MD5) and SHA256 algorithm. In *Journal of Physics: Conference Series*, volume 978. IOP Publishing, 2018.

[15] Ronald L Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[16] Merve Sahin, Aurélien Francillon, Payas Gupta, and Mustaque Ahamad. Sok: Fraud in telephony networks. In *2017 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 235–250. IEEE, 2017.

[17] Adi Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, November 1979.

[18] Rakesh Shrestha, Rojeena Bajracharya, and Seung Yeob Nam. Blockchain-based message dissemination in vanet. In *2018 IEEE 3rd International Conference on Computing, Communication and Security (ICCCS)*, pages 161–166. IEEE, 2018.

[19] Sarah Underwood. Blockchain beyond bitcoin. *Commun. ACM*, 59(11):15–17, October 2016.