A Survey on Financial Exchanges' Cryptographic Protocols, Related Vulnerabilities, and Approaches in DeFi

Xining Li

July 20, 2022

Abstract

In recent years, cybersecurity has been playing a more crucial role in the financial industry and value exchange systems. Accountability and fairness are the most important properties to guarantee a transaction on the platform, which is backed by non-repudiation protocols and fair exchange protocols. We investigated the non-repudiation protocols, fair exchange systems, and Decentralized Finance (DeFi) in digital finance exchanges. The main focus of this survey is the underlying cryptographic protocols used in fintech and e-commerce and vulnerabilities under various implementations and usages. Our methods focus on top literature in non-repudiation, fair exchange, and DeFi protocols. We summarized the literature using formal analysis, the Possum Animation tool to find flaws in non-repudiation and fair exchange protocols. Our results highlighted that many fair-exchange and non-repudiation protocols are flawed, especially the optimistic protocols, few real-life protocols are optimistic, and DeFi is building the non-repudiation based on blockchain and fair exchange based on smart contracts. We concluded that optimistic property is not only possibly flawed but also not a desirable property for companies like Amazon. Our contribution is that we pointed out the issues in optimistic protocols and explained how DeFi achieves non-repudiation and fair exchange.

1 Introduction

Every year, the development of technology in the field of finance is rising more rapidly. It is shown on Statista that 12,216 billion U.S. dollars have been transferred in e-commerce globally in the year 2019 [1]. The average transfer value is 5.04 million U.S. dollars on financial exchanges, and on average, 769,663 transactions are processed per day [2].

As opposed to physical trade, buyers and sellers in e-commerce must have a trust system because the evil human instinct is to get the merchandise without paying or get paid without delivering the merchandise [3] or repudiate the fact of past transactions [4].

In this paper, we had looked into types of non-repudiation and fair exchange protocols such as Optimistic Fair Exchange [3], Optimistic Non-repudiation [5], Semi-Trust by Distributed TTP [6], Multi-Party [7], and No-TTP [8]. We especially focused on the prominent research papers including Zhou and Gollmann's fair non-repudiation protocol based on TTP in 1996 [4], Asokan's optimistic protocols for fair exchange in 1997 [9], Ateniese's Distributed Certified E-Mail Scheme in 2001, Bitcoin in 2008 [10], and FairSwap in 2018 [11]. Details are discussed in the related work section.

Some past surveys on Non-Repudiation [12,13] and Fair Exchange [12,14,15] that also particularly summarized the upper-mentioned research papers are detailed but have not included the approaches in DeFi. The key difference between our survey and the previous surveys is that our survey not only includes the traditional Non-Repudiation and Fair Exchange protocols but also approaches in DeFi.

Our methods are based on the most prominent papers in the fields of non-repudiation, fair exchange, and DeFi protocols. We examined the evolution of non-repudiation, fair exchange, and DeFi protocols by looking at major scientific breakthroughs in chronological order. We made a table addressing the literature using Formal analysis, the Possum Animation Tool that found flaws in non-repudiation and fair exchange protocols.

Our results have shown that many fair-exchange and non-repudiation protocols, particularly optimistic protocols, are problematic; few real-world protocols are optimistic; DeFi is developing non-repudiation based on blockchain and fair exchange based on smart contracts.

We concluded that optimistic property is likely flawed in many non-repudiation and fair-exchange protocols and undesirable for e-commerce like Amazon. Our contribution highlights the following aspects:

- Addressed the cryptographic flaws in optimistic protocols.
- Described how DeFi achieves non-repudiation and fair exchange.
- Discussed what system designers should consider when developing non-repudiation and fair exchange protocols.

2 Related Work

2.1 Past Surveys

Past surveys looked at the different fair exchange techniques that had been invented. Alotaibi et al. [14] categorized the Fair Exchange Protocols into two categories: protocols based on Trusted Third Party (TTP) and protocols without a TTP. In the first category, they summarized protocols that involve a TTP based on inline TTP, online TTP, or offline TTP. The non-TTP-based protocols are mostly Gradual Exchange Protocols. They've summarized the ideas of exchanging secret keys [8] and a randomized protocol for signing contracts [16] but not the Non-repudiation property. Nenadi et al. [17] summarized the enterprise usage of the Non-Repudiation and Fairness in e-commerce applications, but not the details of cryptography. Kremer et al. [12] conducted an intensive literature review on Fair Non-Repudiation Protocols. They also analyzed the leading research in Fair Exchange. However, the paper was published in 2002. Before this, DeFi had not been invented. Blockchain [10] and FairSwap [11] are two important technologies in DeFi. Blockchain provides non-repudiation [10] and FairSwap provides fair exchange [11].

2.2 Our Survey vs. Past Surveys

The financial cryptographic protocols mostly center on non-repudiation and fair exchange. With the emerging DeFi, the science of cryptographic protocols continues to evolve. There are some past surveys on Non-Repudiation [12,13] and Fair Exchange [12,14,15] are informative and detailed, but they have not addressed the approaches in DeFi. Although some papers from top sources work to solve the Fair Exchange issues in DeFi [11], there are no survey papers in Fair Exchange/Non-repudiation this area addressed to the DeFi.

3 Background

While it seems necessary to define all the notions and characteristics of the topics Non-repudiation, Fair Exchange, and DeFi that will be addressed in this paper, the definitions of Non-repudiation and Fair Exchange differ in various literature. DeFi is a broad topic. Due to the limited scope of this paper, we have provided succinct introductions for the concepts of Non-repudiation, Fair Exchange, and DeFi.

3.1 Non-Repudiation

NIST [18] defines Non-repudiation as an "Assurance that the sender of information is provided with proof of delivery and the recipient is provided with proof of the sender's identity, so neither can later deny having processed the information." Hence, fairness isn't a requirement for non-repudiation protocols, but it is a desirable property. The details of the definition is in the Appendix A.1.2.

Non-repudiation epitomizes accountability which is the most important aspect of financial exchanges. Yet, the digital financial system has the two problems below:

- The communication channel is unreliable.
- A communicating party does not play fair or does not follow the protocols.

To tackle the problems above, Brickell et al. [19], in 1987, proposed the Gradual and Verifiable Release of a Secret to prevent possible cheating. However, this design has a high communication overhead. For less communication overhead, Zhou and Gollmann [4] proposed a fair non-repudiation protocol based on TTP in 1996. In this Design, TTP does not know the content of the data, but only the signature of the data and its only responsibilities are to notarize message keys on demand.

Researchers had worked on a multi-party generalization [7], yet there are security vulnerabilities [20].

3.2 Fair Exchange

Fair Exchange can be trivially achieved if all sending parties send their items to a TTP, and the TTP verifies all the items and sends them to the receiving parties. However, in this trivial design, the TTP is the communication bottleneck therefore inefficient. Also, malicious TTPs can subvert participants' assets.

One might think we can use escrow services similar to Amazon. For buying physical goods, escrow services seem necessary. However, these escrow services have known to abuse user information [21]. For purchasing digital goods, maybe a less centralized and semi-trusted third party with the ability to reconstruct the signatures is more desirable [22].

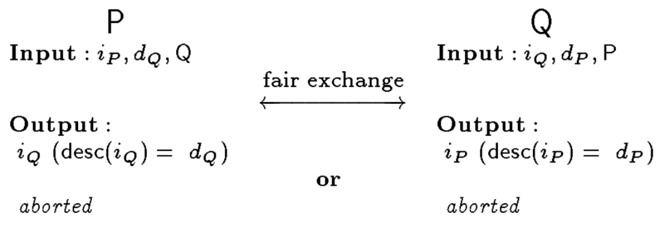

Figure 1: A Successful Fair Exchange [9]

Figure 1 illustrates the flow diagram of a successful exchange that satisfies the following requirements.

Fairness states that if A terminates the protocol in a state where A has B's item, then when B terminates the protocol, B has A's item, and vice versa. This property is often referred to as strong fairness [23, p. 112]. Timeliness states that any honest participant can terminate the exchange unilaterally, i.e., without any help from the opponent. Timeliness guarantees that none of the participants can arbitrarily force the other one to wait for the termination of the exchange [23, p. 112]. The detail definition is in Appendix A.1.1.

Researchers also had contributed intensive efforts to generalize the Fair Exchange to Multi-Party [24–27], yet there are many inherent design flaws [25, 28].

Another direction is to design a semi-trusted TTP [22] mentioned above. Ateniese et al. [6] designed a distributed TTP protocol. Because the TTP duties are distributed, few individual nodes can be malicious without jeopardizing the TTP's total trust [6]. Park et al. [29] improved the verifiable encryption of Ateniese by splitting keys and zero-knowledge proof. Another idea is to adopt a weaker fairness protocol (not even weak fairness) and remove the TTP. Syverson devised a scheme to offer incentives so that principals acting in their own best interests have more motivation to follow the protocol than to abandon it [30]. Leslie Lamport proved that at most $\frac{1}{3}$ of Byzantine-faulty nodes can be tolerated [31]; hence strong fairness without TTP is not possible.

3.3 DeFi

Decentralized Finance (DeFi) is blockchain-based without central intermediaries but instead uses smart contracts. It is a technology that was popularised by Bitcoin. We will briefly discuss the following properties in DeFi.

The main goal of DeFi is to remove the centralized trust. The first successful payment system without a centralized trust is Nakamoto's Bitcoin [10].

In the Results part of this paper, we discuss how blockchain is used to achieve non-repudiation and fair exchange.

4 Method

Our primary research questions(RQ) in this work are: ***RQ1:*** *What are the cryptographic flaws in non-repudiation and fair-exchange protocols?* ***RQ2:*** *How Non-repudiation and Fair Exchange can be achieved in DeFi?* ***RQ3:*** *What are major security breaches and potential vulnerabilities in the real-world implementations of non-repudiation, fair exchange, and DeFi.*

To answer these research questions, we conducted a review of scholarly journals from the top publications (e.g. ACM Symposium on Computer and Communications Security, IEEE Transactions on Information Forensics and Security, and CRYPTO) ranked based on the h5-index in the cybersecurity [32], using keyword "financial exchange", "fair exchange", "financial backend system security", "non-repudiation", "bitcoin", and "fair exchange in DeFi". Through this process, we learned common issues, problems, and their solutions. Therefore we decided to focus on this academic field. We have the following steps:

Step 1: In the screening phase, we surveyed 50+ research papers and came down to a topical research domain.

Step 2: Then, we focused on the early paper that built the foundation of non-repudiation and fairness exchange protocols [4, 9, 22]. We surveyed significant fairness and non-repudiation protocols [17, 24, 33–36], read the definitions of "fairness" and "non-repudiation" from different venues. Through this process, we understood the value exchange protocol deeper.

Step 3: Next, we looked at the significant scientific breakthrough of protocols in chronological sequence to see how non-repudiation, fair exchange, and DeFi protocols have evolved.

Step 4: We summarized the literature using Formal analysis, the Possum Animation Tool to find flaws in non-repudiation and fair exchange protocols.

Step 5: Moreover, we looked at relevant attacks and vulnerabilities related to real-world financial exchanges and analyzed those using the STRIDE model, where we found attack events on SWIFT.

Step 6: Finally, we summarized the approaches of fair exchange and non-repudiation in DeFi.

We include the following items as our scope:

- Influential optimistic non-repudiation protocols [4, 5, 7, 37].
- Influential fair exchange protocols [13, 38–40].
- Cryptographic flaws in non-repudiation and fair-exchange protocols found by Formal analysis and Possum Animation Tool (For RQ1)
- Bitcoin escrow transaction, ZKCP, and FairSwap (For RQ2)
- SWIFT and SIMAP protocol (RQ3)

5 Results

This section reviews most of the non-repudiation and fair exchange protocols in our cited resources. We give a comparison in the following table, where we summarize essential information such as whether those protocols failed against cryptographic adversary models and which paper proposed those adversary models and conducted the underlying attacks.

	formal analysis issues	Possum Animation	STRIDE	type
Zhou-Gollmann protocol [4]	failed [35, 41, 42]			optimistic
Mukhamedov multi-party fair exchange [38]	failed [43]			Multi-party
Kremer-Markowitch Two-Party Non-Repudiation Protocol [5]	passed [42]			optimistic
Kremer Multi-Party Optimistic Non-Repudiation Protocol [7]	failed [42]	failed [20]		optimistic multi-party
Gomila and Rotger' Asynchronous Protocol for Optimistic Certified Electronic Mail [37]		failed [20]		optimistic
Zhou, Deng, and Bao Fair Exchange Protocol [13]		issues found [20]		
Asokan, Shoup, and Waidner Fair Exchange Protocol [39]	failed [44, 45]	failed [20]		optimistic
Baum-Waidner Multi-Party Contract Signing Protocol [40]	passed [42, 46]			multi-party
SWIFT [47]	source not open	source not open	Repudiation	implementation
SIMAP [48]	untested	untested	Spoofing Possibility	implementation
FairSwap [11]	passed [11]	untested	Spoofing Possibility	implementation

Table 1: Result Summary

Table 1 includes three methods of security analysis which are defined as follows.

- **Formal analysis** is a method that abstracts the cryptographic protocols into mathematical structures and uses formal proofs to verify the cryptographic protocols. CafeOBJ is a most advanced formal specification language which inherits many advanced features (e.g. flexible mix-fix syntax, powerful and clear typing system with ordered sorts, parameteric modules and views for instantiating the parameters, and module expressions, etc.) from OBJ (or more exactly OBJ3) algebraic specification language [49].

- **The Possum Animation tool** is a software tool created by the University of Queensland's Software Verification Research Centre. It allows developers to visualize specifications expressed in SUM, which is effectively a markup for the specification language Z [50]. In the context of formal specification, animation allows the specifier to ask questions about their developing specification that are automatically answered. In contrast, formal proofs are frequently used to demonstrate that specifications contain the desired qualities. At several stages of the formal development process, animation provides an appealing, low-cost alternative to formal evidence [20, 51].

- **STRIDE** is a model for identifying computer security threats, providing a mnemonic for security threats in the following categories [52]: Spoofing, Tampering, Repudiation, Information disclosure (privacy breach or data leak), Denial of Service, and Elevation of privilege.

In Table 1, we aggregated the non-repudiation and fair protocols, and we found that many optimistic protocols have flaws. We found most of the multiparty optimistic protocols have failed the formal analysis issues, including Kremer Multi-Party Optimistic Non-Repudiation Protocol [7] and Mukhamedov multi-party fair exchange [38]. Most optimistic protocols have failed Possum Animation tools [20] in Table 1.

In Table 1, Gurgens, Kremer, and Boyd are 3 of the researchers that found most of the flaws.

Kremer found flaws in the Zhou-Gollman protocol and Kremer Multi-Party Optimistic Non-Repudiation Protocol he proposed earlier [42]. Gurgens proposed several attacks on Non-Repudiation Protocols [41]. In particular, Gurgens et al. studied the cryptanalysis of Zhou-Gollman protocol from other researchers that approved the Zhou-Gollman protocol. Using formal analysis, Gurgens et al [35]. found that in the Zhou-Gollman protocol, the security goal NRO(Alice) (Non-repudiation of Origin and Receipt of the participant Alice) is not satisfied.

Boyd et al [20] use the software called The Possum Animation Tool. They found various issues in the following protocols:

1. Kremer Multi-Party Optimistic Non-Repudiation Protocol [7]

2. Gomila and Rotger' Asynchronous Protocol for Optimistic Certified Electronic Mail

3. Zhou, Deng, and Bao Fair Exchange Protocol

4. Asokan, Shoup, and Waidner Fair Exchange Protocol

We were unable to find a real-world application of optimistic protocols, but in the remaining part of this section, we discuss the real-world example of failing to achieve the non-repudiation requirement and the real-world example of multi-party fair exchange protocol that might fail its business target.

5.1 Real-World Implementation of Non-Repudiation

One famous real-world implementation is the SWIFT network, which is an efficient way to transfer large data files that offer non-repudiation [47]. FileAct takes advantage of the SWIFTNet Public Key Infrastructure (PKI) [53], which allows for file authentication and integrity management. It also has a strong registration mechanism for certificate issuance and maintenance [47]. The nonrepudiation of file transfer and receipt is guaranteed by the PKI signature. FileAct maintains a centralized log of all files that are transmitted. SWIFT can provide independent evidence proving that the transfer occurred as claimed in the event of a dispute [47].

However, the possibility of private key theft still exists for the non-repudiation protocol. Therefore, the SWIFT Qualified Certificates defines a detailed procedure for revoking a certificate. At any time, the Subscriber can revoke its SWIFT Qualified Certificate [54].

5.2 Root CA Key Security Analysis

Assuming the cryptographic protocol used for non-repudiation is correct, the security of the SWIFT network relies mostly on the SWIFTNet Certification Authority (CA) [47,54], which is a validator that validates the identities of subscribers [54]. Once the Root Key is compromised, the attacker can issue fraudulent certificates at will, pretending to be any company/institution such as Chase, RBC, BOA, etc., directly subverting the credit system of the entire Financial System. Users will not be able to distinguish whether the opposite side of the network cable is a bank or a hacker.

5.3 SWIFT Security Breach

According to a New York Times report in 2016 [55], an $81 million loss from the Bangladesh central bank via its account at the New York Federal Reserve Bank was attributed to the SWIFT's Alliance Access software hack [55]. The malware used in the hack was designed to send unauthorized SWIFT communications while also concealing the fact that they had been delivered. In other words, it is a repudiation attack. The virus wiped the database record of the transactions after sending the SWIFT messages that stole the funds and then took additional steps to prevent confirmation messages from revealing the theft [55].

Figure 4 in the F-Secure report shows several high-profile SWIFT-related attacks. These threat actors have frequently utilized specifically designed malware and innovative strategies to achieve their goal of completing fraudulent money transactions in some of the most high-profile attacks. Therefore, the tampering of the SWIFT network is mostly a result of impersonation. However, the lack of backup caused the repudiation to be possible. SWIFT has responded by implementing CSP to defend its global SWIFT community from this threat [53].

5.4 Fair Exchange Results

A transaction with fair exchange entails two parties fulfilling their duties; a contract outlines the consequences if one of them fails to do so. Purchasing things, for example, entails the merchant delivering the item and the buyer paying for it at the same time. Often fair exchange protocols are the backbone of the fair non-repudiation protocols, and the applications include key exchanges, e-books, software licenses, music, etc. Fair exchange is crucial for exchanges and e-commerce, and improper usage and attack on fair exchange protocols can lead to severe consequences. Secure multi-party computation is a similar problem to multi-party fair exchange. Secure multi-party computation aims to create methods for parties to jointly compute a function over their inputs while keeping those inputs private. Kılınç et al. [56] proved that fair and secure multi-party computation can be achieved by applying a multi-party fair exchange protocol to any secure multi-party computation protocol.

We have seen a trend that multi-party fair exchange protocols can help solve various problems in multilateral negotiations, and we have addressed the issues of multi-party fair exchange protocols in this section.

5.5 Real-Life Deployment of Multi-Party Fair Exchange Protocols

Bogetoft et al. [48] used a multi-party computation to achieve a multi-party fair exchange, which is used in multilateral negotiation. Multi-party fair exchange protocols can help to achieve convenience in multilateral negotiation [48]. Few multi-party computation protocols have been deployed [48], mainly because of the lack of understanding in the general public of the potential of the technology [48].

We found one real-world multi-party fair exchange protocol implementation, the Danish Sugar Beet Auctions' SIMAP [48]. The project has developed and launched a Secure multi-party computation protocol with fair exchange support for sugar farmers to set the market-clearing price, sponsored by Danish

Strategic Research Council and formed by the Secure Computing, Economy and Trust, Copenhagen Business School, and Secure Information Management and Processing [48]. Several thousand farmers produce sugar and sell them on Danisco, where multilateral negotiations for the market-clearing price are often needed and which could have led to lengthy discussions.

From the academic perspective, SIMAP [48] assumes that the servers have set up public-secret key pairs before the computation starts. The public keys are available to the clients. All servers' secrets share a random value and add all shares locally to share a random unknown number. Through some one-way functions, the protocol will produce the random shared binary value. The shared binary value will then be stored in the shared-mind virtual machines and used for encrypting the bids.

The auction implementation is as follows: The party $P1$ receives the bids encrypted from the bidders. However, the bids are encrypted with the public key of another party, $P2$. Then $P1$ sends the encryptions, randomized and in permuted order to $P2$ that decrypts the bids. This approach achieves security because $P2$ does not know who placed which bids. Through a comparison protocol, all bit-pairs will be computed [48].

After the deadline for the auction had passed, the servers were connected to the database and each other, and the market-clearing price was securely computed, as well as the quantity each bidder would buy/sell at that price [48].

The cryptographic framework of SIMAP is shown in the Figure 5.

However, there are some security concerns on this innovative technology

- While Bogetoft et al. [48] proved lemmas and theorems used in this paper, there is no security proof of fair exchange using formal analysis. Also, the SIMAP is not open source, violating the Open-Design principle.

- SIMAP does not explicitly implement any security against cheating bidders, other than verifying their identity [48].

5.6 Approaches in DeFi

The previous sections show that trust is a concern on the TTP, and researchers have proposed semi-trusted TTP [22] and distributed TTP [29]. Blockchain is a distributed trust technology, and DeFi uses blockchain as the TTP to resolve the two-party non-repudiation and fair exchange.

5.6.1 Non-Repudiation

Bitcoin uses UTXO (Unspent Transaction Output) model [10]. In this model, individual coins (UTXOs) are transferred between users. The payer transfers a coin to the payee by digitally signing a hash of both the previous transaction and the public key of the payee. Then the payer would add the MAC to the end of the coin. A payee can verify the signatures to verify the chain of ownership.

To prevent tamper on the transactions, bitcoin uses Merkle Tree [57] in which transaction ids are hashed together on a block, passing to the parent nodes, and the parent nodes are hashed together and advanced to the grandparent nodes all the way to the Merkle root (the block header). Any changes on the transaction will be cascade to the Merkle root hence invalidating the block, and other nodes will ignore the invalid block (illustrated in the Figure 2) [10]. We can see that bitcoin satisfies the Non-repudiation property as anyone on the network can verify the evidence of a transaction.

5.6.2 Fair Exchange

It is worth noticing that in 2015 Smith [58] proposed a method called escrow transaction in Bitcoin. In such a transaction, there are three parties: Alice (the buyer), Bob (the seller), Judy (the judge). In such an exchange, Alice puts some coins in a multi-sig transaction with an expiration date on the blockchain, which requires two or three people to sign to redeem the coins [58]. The judge will only engage in resolving a dispute between the buyer and seller. For most of the time, the judge will not need to engage in the transactions.

While the Escrow via Multisig is optimistic [58], the fairness is entirely based on the mediator. The protocol itself will not ensure fairness. By definition, in section 3.2, Escrow via Multisig is not a fair exchange protocol.

While it may be necessary to have an escrow service anyway if the transaction is to buy physical goods, fairness in good digital transactions can be achieved by blockchain automatically reconstructing and verifying the signatures.

To give the blockchain the ability to reconstruct the signatures, Maxwell completed the first successful Zero-Knowledge Contingent Payment (ZKCP) on the Bitcoin network in 2016 [59].

However, Maxwell's ZKCP protocol [59] was broken by Campanelli et al. [60] in 2017.

In 2018 Dziembowski et al. constructed an efficient protocol for fair exchange of digital goods using smart contracts called FairSwap [11] with provable security.

The FairSwap protocol guarantees strong fairness [11]. In FairSwap, the smart contract would first get the hash in the money, store the money, then receive the file. The buyer would now be able to see the file (the blockchain is transparent) and read it from the smart contract [11]. The smart contract will then hash the big file and verify if it's the correct hash if this is the case the seller gets the payment if this is not the case the buyer gets the money back [11].

The figure 3 shows that the sender S sells a file to receiver R via the judge implemented with Smart Contract. During the protocol execution, the judge takes the key from the sender and the coin from the receiver. FairSwap provides the security proof and resolves the inefficiency for large files issue in ZKCP by hash functions. However, the FairSwap protocol is still far from an optimistic fair exchange protocol because it needs to interact with the blockchain during opening and closing [11].

6 Discussion

The main motive of this section is to discuss how the non-repudiation protocols and fair exchange protocols reached this status quo? Why we do not see large-scale real-world implementations of optimistic protocols?

For fair exchange and non-repudiation protocols, it is believed that the TTP could become a bottleneck. In 2005, Nenadic et al. [36] suggested that the optimistic approach should be the direction for the design of fair exchange solutions as this approach reduces the possibility for the TTP to become a performance and security bottleneck, and it reduces the level of trust placed upon the TTPs.

Therefore, designers attempt to remove the centralized accountability of the TTP. However, optimistic protocols are much more complicated. Researchers found issues in optimistic fair exchange protocols and optimistic fair non-repudiation protocols [20, 35, 41–43]. Kremer himself did the formal analysis on his Kremer-Markowitch two-party non-repudiation protocol [42]. However, no other researchers have used different approaches (e.g. the Possum Animation tool [20]) to support his claim. Maybe because of this, his two-party optimistic non-repudiation protocol [5] has not been widely accepted.

Therefore, it is reasonable for FairSwap to implement the smart contract as the Online TTP in which the TTP is involved in every protocol transfer. In an attempt to find the real-world implementations of optimistic fair exchange protocols and optimistic non-repudiation protocols, we realized there are no implementations on the market claiming to use optimistic protocols (OptiSwap is still under development).

Neumann said, [61] "If you think cryptography is the answer to your problem, then you don't know what your problem is." From our observations, escrow services' non-repudiation and fairness should not and in fact do not depend on cryptography; customers using goods online do not care about efficiency in the back-end servers; escrow service providers can also make a profit from product recommendations to users by the recommendation systems [62, 63]. Amazon, Taobao, and JingDong epitomize escrow services. In such e-commerce platforms, the goods of the transactions are verified and approved by the customer services, not cryptography. In case of dispute, the customer service would have the authority to roll back the money paid by the client.

In the Results section, we have shown that some non-repudiation and fairness cryptographic protocols have flaws. We believe that optimistic protocols such as Zhou-Gollmann [4] are broken because of the incapability to prove that incorrect messages have been sent to the TTP. Multi-party protocols such as Kremer Multi-Party Optimistic Non-Repudiation Protocol [7] failed formal analysis because of improper resolution of the Byzantine General Problem.

Unsurprisingly, we have not seen a scalable real-world application using optimistic protocols. The optimistic protocols are rarely used, possibly even rarely used secretly in safety-critical back-end systems because of their complexity and possible cryptographic attacks.

We have seen that to effectively achieve fairness in exchanges, the e-commerce industry's approach is to build a more powerful TTP and increase the efficiency of the TTP by finding a balance between Consistency, Availability, and Partition tolerance for different use cases [64–66]. Another reason for building the centralized TTPs for fair exchange is that companies maintaining the centralized TTPs can put the TTP exchanged information in the Extract Transform Load (ETL) pipelines of the recommendation systems [62, 63].

Optimistic protocols would prevent the e-commerce industry from integrating recommendation systems into the transactions, trivially because the backend servers implementing the optimistic protocols do not need to engage in the transactions. Therefore, we believe that the e-commerce industry lacks financial incentives to develop the science and technology for optimistic protocols. Maybe because of this, the real-life examples we provided in this paper do not use optimistic protocols.

Nevertheless, it is not to say that implementations of non-optimistic protocols are secure. Designing secure systems need much more than the cryptographic algorithms, and we have found SWIFT was attacked not on its cryptographic algorithms but on the side-channel opened by malware, and its non-repudiation is compromised because of the lack of the backup of its database. The SIMAP (in Figure 5) project's economic implication is remarkable. However, as a national research project supported by the Danish Strategic Research Council, the SIMAP project is not open source, violating the open-design principle. Moreover, there is no research about its related attacks as today. Due to its complexity, users may fail to fulfill the security requirements making spoofing a possibility.

7 Conclusion

Throughout the paper, we surveyed most of the existing non-repudiation protocols. At the beginning of the paper, we defined the properties of non-repudiation and fair exchange protocols. Then we browsed through the different approaches of the non-repudiation and fair exchange protocols and showed the evolution of the involvement of the TTP from protocols.

Finally, we summarized the existing methods for formally verifying non-repudiation and fair exchange protocols. From our studies on the past research, we summarized that many optimistic protocols are flawed and are rarely implemented in the real world. Many multi-party fair exchange protocols are also flawed. Currently, there is no large-scale application of a multi-party fair exchange protocol. In real-world implementations, non-repudiation is often breached by the side-channel attack. To address the issue of the non-repudiation attack, we summarized the circumstances and vulnerabilities underlying the SWIFT attack. Admittedly, due to the vast scope of this topic, many protocols and approaches are not covered, but we hope this paper can raise the awareness of financial system designers and provide them a starting point to learn about non-repudiation and fair exchange protocols.

A APPENDIX

A.1 Definitions

A.1.1 Fair Exchange

In this optimistic fair exchange protocol, the TTP is only used if the participants misbehave. Asokan formulates three important requirements of fair exchange [9]. $desc()$ is a function mapping exchangeable items to underlying details in the transaction. P initiates the exchange with an item i_P and a description d_Q of the expected item (usually a key). For Q, a similar definition can be stated.

The requirement of P is as follows:

- *Effectiveness*: If Q also behaves correctly and both P and Q do not abort the protocol, then when the protocol has completed P has i_Q such that $desc(i_Q) = d_Q$.
- *Fairness*:
  - *Strong Fairness*: When the protocol has completed, either P has i_Q such that $desc(i_Q) = d_Q$, or Q has no additional information about i_P
  - *Weak Fairness*: When the protocol has completed, either it satisfies the *Strong Fairness*'s requirement or P can prove to the TTP that Q has received i_P such that $desc(i_P) = d_P$
- *Timeliness*: both parties can abort the exchange unilaterally without degrading the fairness achieved by P

The requirement of Q can be stated in a trivially similar manner [9]. It can be trivially inferred that Weak Fairness implies the need for Non-repudiation property while Strong Fairness depending on the context, might not need Non-repudiation.

A.1.2 Non-Repudiation

The basic requirement of fair non-repudiation is as follows: Suppose Alice wants to transfer a message to Bob. **EOO**(Evidence of origin for Bob): proves that Alice has sent message M

EOR(Evidence of receipt for Alice): Proves that Bob has received message M

A Fair Non-Repudiation Protocol has the following properties [41]:

- Fair non-repudiation of **receipt**: When sender completes the protocol, it is
  - Either sender owns a valid evidence of receipt created by receiver
  - Or receiver has neither message nor a valid evidence of receipt.
- Fair non-repudiation of **origin**: When receiver completes the protocol, it is
  - Either receiver owns message and a valid evidence of origin created by sender
  - Or sender has no valid evidence of receipt.
- Timeliness for sender and receiver: Each party can complete the protocol.

B A Cryptographic Attack on the ZDB Protocol

The ZDB protocol [13] was intended to present a more efficient fair non-repudiation protocol based on the ideas from the original Zhou and Gollman's fair non-repudiation protocol [4].

- Here is the notation of the ZDB protocol [4].
  - M: message being sent from A to B
  - K: symmetric key generated by A.
  - M, N: concatenation of two messages M and N.
  - $H(M, K)$: a one-way hash function applied to messages M and the key K
  - $e_K(M)$ and $d_K(M)$: encryption and decryption of the message M with the key K.
  - $C = eK(M)$: committed ciphertext for message M.
  - $L = H(M, K)$: label to link C and K.
  - $f1, f2, ..., f8$: message flags to indicate the purpose of the respective message.
  - $E_{TTP}(K)$ encryption of key K with TTP's public key.
  - $sig_A(M)$: principal A's digital signature on message M with A's private signature key. Note that the plain text is not recoverable from the signature, i.e. for signature verification the plaintext needs to be made available.
  - $EOO_C = sig_A(f1, B, L, C)$: evidence of origin of C.
  - $EORC = sig_B(f2, A, L, EOO_C)$: evidence of receipt of C.
  - $EOOK = sig_A(f3, B, L, K)$: evidence of origin of K.
  - $EOR_K = sig_B(f_4, A, L, EOO_K)$: evidence of receipt of K.
  - $subK = sig_A(f_5, B, L, K, TTP, EOO_C)$: evidence of submission of K to the TTP.
  - $conK = sig_{TTP}(f_6, A, B, L, K)$: evidence of confirmation of K by the TTP.
  - $abort = sig_{TTP}(f_8, A, B, L)$: evidence of abortion.
  - $A \rightarrow B : M$:Party A sends message M with B being the intended recipient.
- the ZDB protocol consists of the following sub-protocols [41].
  - Exchange sub-protocol
    1. $A \rightarrow B : f_1, f_5, B, L, C, TTP, E_{TTP}(K), EOO_C, sub_K$
    2. IF B gives up THEN quit ELSE $B \rightarrow A : f_2, A, L, EOR_C$
    3. IF A gives up THEN abort ELSE $A \rightarrow B : f_3, B, L, K, EOO_K$
    4. IF B gives up THEN resolve ELSE $B \rightarrow A : f_4, A, L, EOR_K$
    5. IF A gives up THEN resolve
  - Abort sub-protocol which can only be performed by A
    1. $A \rightarrow TTP : f_7, B, L, sig_A(f_7, B, L)$
    2. IF resolved THEN $TTP \rightarrow A : f_2, f_6, A, B, L, K, con_K, EOR_C$
    3. ELSE $TTP \rightarrow A : f8, A, B, L, abort$
  - Resolve sub-protocol
    1. $U \rightarrow TTP : f_1, f_2, f_5, A, B, L, TTP, E_{TTP}(K), sub_K, EOO_C, EOR_C$
    2. IF aborted THEN $TTP \rightarrow U : f_8, A, B, L, abort$

3. ELSE $TTP \rightarrow U : f2, f6, A, B, L, K, conK, EORC$

However, Gurgens accurately points out that in the ZDB model mentioned above, it is possible for the sender to send the wrong sub_K or $E_{TTP}(K)$ at the first step of the exchange [41]. Sadly, *Bob* cannot prove this misbehavior of *Alice*. Then the resolve sub-protocol stops with an error when started by B. This again prevents B from terminating. However, A can construct a resolve request with the correct encryption of the key, which allows A to complete the protocol at any time. Later in Gurgens's review [41], he proposed an improved definition of messages by using a hash on the ciphertext of encryption of the key, adding the E_K in both EOO_C and EOR_C. His modification has been approved by various literature [34, 67, 68].

C A Cryptographic Attack on a Multi-Party Fair Exchange Protocol

An n-party multi-unit general exchange is defined by a n by n matrix E such that the vector E_{ij} representing a basket of goods given by the party P_i to the party P_j [24].

Macià et al. [43] described a cryptographic replay attack on the multi-party fair exchange protocol proposed in Mukhamedov's Multi-party Fair Exchange Protocol [38].

Mukhamedov's Multi-party Fair Exchange Protocol is defined as follows [38, 43]:

- Each participant knows the identity of the remaining participants in the exchange;
- All participants agree on the TTP and the functions f and F_n. Here f is a homomorphic one-way function. $F_n(x_1, f(x_2), ..., f(x_n)) = f(x_1 \cdot x_2 \cdot ... \cdot x_n)$
- Descriptions of the items to be exchanged $f(m_i)$ are public.

The label l has been added to identify the run of a protocol. To minimize label space collisions, Mukhamedov et al. explain that during the setup phase, the TTP will generate and send the value of l to the exchange participants (timestamps or nonces, together with the TTP's name, may suffice). The following is the protocol:

The protocol is as follows:

- $P_i \rightarrow P_{i+1} : PR_{Pi}(l, PU_{Pi+1}(R_i))$
- $Pi \rightarrow TTP : PR_{Pi}(l, A_i, PU_T(C_i), f(R_i), f(R_{i-1}), f(m_i - 1))$
- $TTP \rightarrow Pi : PR_T(l, C)$, where $C = \{C_i | 1 \leqslant i \leqslant n\}$

The notations are defined as follows:

- $A_i = F_n(m_i, f(m_1)...f(m_{i-1}), f(m_{i+1})...f(m_n))$ and $C_i = m_i \cdot R_i^{-1}$
- $PU_A(m)$ is the result of applying an asymmetric encryption algorithm to the plaintext m under sender's public key;
- $PR_A(m)$ denotes the digital signature of sender over message m using her private key;
- $E_K(m)$ is the symmetric encryption of message m under key K.

In the protocol described above, a replay attack on the TTP might be possible. A group of dishonest participants deceives the TTP into believing that the transaction has been halted. Then, without an honest participant from the prior trade, they establish a new protocol and utilize the TTP to retrieve the item of this participant from the first transaction [43].

Macià et al provide the following example, which has the following assumptions.

- three participants formed a cyclic topology using the Mukhamedov's Multi-party Fair Exchange Protocol [38,43]. And they assume that the participant P_1 is colluding with P_2 against P_3.
- P_1 and P_2 capture the message sent by P_3 to the TTP. In particular, the message they will capture is as follow: $PR_{P3}(l, A_3, PU_T(C_3), f(R_3), f(R_2), f(m_2))$
- P_1, P_2, and P_4 (a new dishonest participant who collaborated with P_1 and P_2) then begin a new procedure run. The target of this new exchange is getting the item m_3 of P_3 from the previous exchange. In this case, P_1 and P_2 are supposed to exchange new items (the description of the items are $f(\hat{m}_1)$ and $f(\hat{m}_2)$ respectively). But these participants deceive the TTP, they say that the description of the item exchanged by P_4 is $f(m3)$. In particular the second step will be as follows:
  - $P1 \rightarrow TTP : PR_{P1}(\hat{l}, \hat{A}_1, PU_T(\hat{C}_1), f(\hat{R}_1), f(R_3), f(m_3))$
  - $P2 \rightarrow TTP : PR_{P2}(\hat{l}, \hat{A}_2, PU_T(\hat{C}_2), f(\hat{R}_2), f(R_1), f(\hat{m}_1))$
  - $P4 \rightarrow TTP : PR_{P4}(\hat{l}, \hat{A}_1, PU_T(\hat{C}_3), f(R_3), f(\hat{R}_2), f(\hat{m}_2))$

Later, Macià et al. [43] proposed a fix to the the Mukhamedov's Multi-party Fair Exchange Protocol [38,43].

We see that the multiparty fair exchange protocols are difficult to design and analyze mainly because in the Byzantine failure model [31], a faulty participant may deviate from the fair exchange protocols in any fashion, and parties can collude with other parties.

D Images

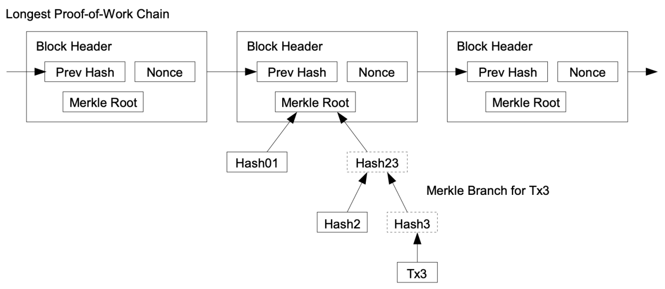

Figure 2: Bitcoin Structure [10]

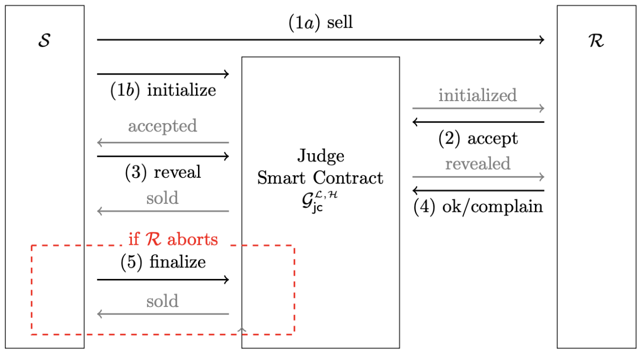

Figure 3: Outline of fair exchange with judge contract [11]

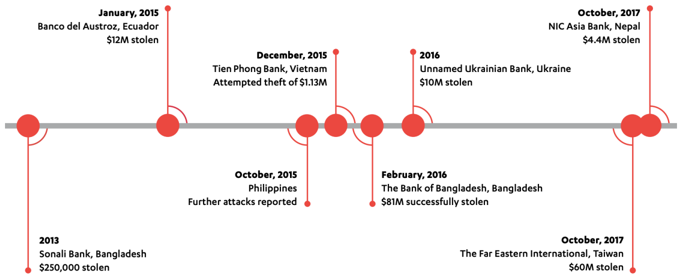

Figure 4: Timeline: High-profile SWIFT-related attacks [53]

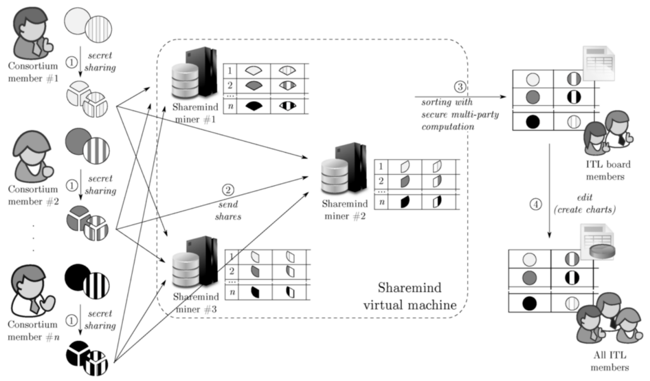

Figure 5: SIMAP architecture [69]

References

[1] D. Coppola, "Global b2b e-commerce gross merchandise volume (gmv) from 2013 to 2019," 10 2021. [Online]. Available: https://www.statista.com/statistics/705606/global-b2b-e-commerce-gmv

[2] "The federal reserve, monthly fedwire funds service statistics." [Online]. Available: https://www.frbservices.org/resources/financial-services/wires/volume-value-stats/monthly-stats.html

[3] N. Asokan, M. Schunter, and M. Waidner, "Optimistic protocols for fair exchange," in *Proceedings of the 4th ACM Conference on Computer and Communications Security*, ser. CCS '97. New York, NY, USA: Association for Computing Machinery, 1997, p. 7–17. [Online]. Available: https://doi.org/10.1145/266420.266426

[4] J. Zhou and D. Gollman, "A fair non-repudiation protocol," in *Proceedings 1996 IEEE Symposium on Security and Privacy*, 1996, pp. 55–61.

[5] S. Kremer and O. Markowitch, "Optimistic non-repudiable information exchange," in *Symposium on Information Theory in the Benelux.* Citeseer, 2000, pp. 139–146.

[6] G. Ateniese, B. de Medeiros, and M. T. Goodrich, "Tricert: A distributed certified e-mail scheme," in *NDSS*, 2001.

[7] O. Markowitch and S. Kremer, "A multi-party optimistic non-repudiation protocol," in *Proceedings of the Third International Conference on Information Security and Cryptology*, ser. ICISC '00. Berlin, Heidelberg: Springer-Verlag, 2000, p. 109–122.

[8] M. Blum, "How to exchange (secret) keys," *ACM Trans. Comput. Syst.*, vol. 1, no. 2, 1983.

[9] N. Asokan, V. Shoup, and M. Waidner, "Asynchronous protocols for optimistic fair exchange," in *Proceedings. 1998 IEEE Symposium on Security and Privacy (Cat. No.98CB36186)*, 1998, pp. 86–99.

[10] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2009. [Online]. Available: http://www.bitcoin.org/bitcoin.pdf

[11] S. Dziembowski, L. Eckey, and S. Faust, "Fairswap: How to fairly exchange digital goods," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '18. New York, NY, USA: Association for Computing Machinery, 2018, p. 967–984. [Online]. Available: https://doi.org/10.1145/3243734.3243857

[12] S. Kremer, O. Markowitch, and J. Zhou, "An intensive survey of non-repudiation protocols," *Computer Communications*, vol. 25, pp. 1606–1621, 11 2002.

[13] J. Zhou, R. Deng, and F. Bao, "Evolution of fair non-repudiation with ttp," in *Information Security and Privacy*, J. Pieprzyk, R. Safavi-Naini, and J. Seberry, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 258–269.

[14] A. Alotaibi and H. Aldabbas, "A review of fair exchange protocols," *International journal of Computer Networks & Communications*, vol. 4, 07 2012.

[15] J.-C. Loh, S.-H. Heng, and S.-Y. Tan, "A survey on optimistic fair exchange protocol and its variants," 05 2017, pp. 1–6.

[16] S. Even, O. Goldreich, and A. Lempel, “A randomized protocol for signing contracts,” *Commun. ACM*, vol. 28, no. 6, p. 637–647, Jun. 1985. [Online]. Available: https://doi.org/10.1145/3812.3818

[17] Z. N. Nenadi A., “Non-repudiation and fairness in electronic data exchange,” *Springer, Dordrecht*, 2004.

[18] C. C. Editor, “Non-repudiation - glossary.” [Online]. Available: https://csrc.nist.gov/glossary/term/non_repudiation

[19] E. F. Brickell, D. Chaum, I. Damgård, and J. van de Graaf, “Gradual and verifiable release of a secret,” in *CRYPTO*, 1987.

[20] C. Boyd and P. Kearney, “Exploring fair exchange protocols using specification animation,” in *Information Security*, G. Goos, J. Hartmanis, J. van Leeuwen, J. Pieprzyk, J. Seberry, and E. Okamoto, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 209–223.

[21] J. Greig, “Amazon fined $887 million for gdpr privacy violations,” 7 2021. [Online]. Available: https://www.zdnet.com/article/amazon-fined-887-million-for-gdpr-privacy-violations/

[22] M. K. Franklin and M. K. Reiter, “Fair exchange with a semi-trusted third party (extended abstract),” in *Proceedings of the 4th ACM Conference on Computer and Communications Security*, ser. CCS ’97. New York, NY, USA: Association for Computing Machinery, 1997, p. 1–5. [Online]. Available: https://doi.org/10.1145/266420.266424

[23] B. Rosenberg, *Handbook of financial cryptography and security.* CRC Press, 2010.

[24] M. Franklin and G. Tsudik, “Secure group barter: Multi-party fair exchange with semi-trusted neutral parties,” in *Financial Cryptography*, R. Hirchfeld, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 90–102.

[25] J. A. Garay and P. MacKenzie, “Abuse-free multi-party contract signing,” in *Distributed Computing*, P. Jayanti, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 151–166.

[26] M. Franklin and G. Tsudik, “Secure group barter: Multi-party fair exchange with semi-trusted neutral parties,” in *International Conference on Financial Cryptography.* Springer, 1998, pp. 90–102.

[27] N. Asokan, M. Schunter, and M. Waidner, “Optimistic protocols for multi-party fair exchange,” 1996.

[28] R. Chadha, S. Kramer, and A. Scedrov, “Formal analysis of multi-party contract signing,” in *Proceedings. 17th IEEE Computer Security Foundations Workshop, 2004.*, 2004, pp. 266–279.

[29] J.-M. J. Park, E. Chong, and H. Siegel, “Constructing fair-exchange protocols for e-commerce via distributed computation of rsa signatures,” vol. 22, 01 2003, pp. 172–181.

[30] P. Syverson, “Weakly secret bit commitment: applications to lotteries and fair exchange,” in *Proceedings. 11th IEEE Computer Security Foundations Workshop (Cat. No.98TB100238)*, 1998, pp. 2–13.

[31] L. Lamport, R. E. Shostak, and M. C. Pease, “The byzantine generals problem,” *ACM Trans. Program. Lang. Syst.*, vol. 4, pp. 382–401, 1982.

[32] “Top publications: Computer security & cryptography.” [Online]. Available: https://scholar.google.ca/citations?view_op=top_venues&hl=en&vq=eng_computersecuritycryptography

[33] O. Markowitch and S. Kremer, "An optimistic non-repudiation protocol with transparent trusted third party," vol. 2200, 08 2002.

[34] J. Feng, Y. Chen, and D. H. Summerville, "A fair multi-party non-repudiation scheme for storage clouds," in *2011 International Conference on Collaboration Technologies and Systems (CTS).* IEEE, 2011, pp. 457–465.

[35] S. Gürgens and C. Rudolph, "Security analysis of (un-) fair non-repudiation protocols," *Formal Aspects of Computing*, vol. 17, pp. 260–276, 01 2005.

[36] A. Nenadic and N. Zhang, *Non-Repudiation and Fairness in Electronic Data Exchange.* Kluwer Academic Publishers, 01 2005, pp. 286–293.

[37] J. Ferrer-Gomila, M. F. Hinarejos, G. Draper Gil, and L. Rotger, "Optimistic protocol for certified electronic mail with verifiable ttp," *Computer Standards & Interfaces*, vol. 57, 11 2017.

[38] A. Mukhamedov, S. Kremer, and E. Ritter, "Analysis of a multi-party fair exchange protocol and formal proof of correctness in the strand space model," vol. 3570, 02 2005, pp. 255–269.

[39] N. Asokan, V. Shoup, and M. Waidner, "Optimistic fair exchange of digital signatures," in *International Conference on the Theory and Applications of Cryptographic Techniques.* Springer, 1998, pp. 591–606.

[40] B. Baum-Waidner and M. Waidner, "Round-optimal and abuse-free optimistic multi-party contract signing," in *Automata, Languages and Programming*, U. Montanari, J. D. P. Rolim, and E. Welzl, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 524–535.

[41] S. Gürgens, C. Rudolph, and H. Vogt, "On the security of fair non-repudiation protocols," in *International Conference on Information Security.* Springer, 2003, pp. 193–207.

[42] S. Kremer, "Formal analysis of optimistic fair exchange protocols," 2004. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.58.9931

[43] M. Mut-Puigserver, M. Payeras-Capellà, J. L. Ferrer-Gomila, and L. Huguet-Rotger, "Replay attack in a fair exchange protocol," in *Applied Cryptography and Network Security*, S. M. Bellovin, R. Gennaro, A. Keromytis, and M. Yung, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 174–187.

[44] P. H. Drielsma and S. Mödersheim, "The asw protocol revisited: A unified view," *Electronic Notes in Theoretical Computer Science*, vol. 125, no. 1, pp. 145–161, 2005.

[45] V. Shmatikov and J. C. Mitchell, "Finite-state analysis of two contract signing protocols," *Theor. Comput. Sci.*, vol. 283, no. 2, p. 419–450, Jun. 2002. [Online]. Available: https://doi.org/10.1016/S0304-3975(01)00141-4

[46] R. Chadha, S. Kremer, and A. Scedrov, "Formal analysis of multiparty contract signing," *Journal of Automated Reasoning*, vol. 36, no. 1, pp. 39–83, 2006.

[47] "Fileact: Swift - the global provider of secure financial messaging services." [Online]. Available: https://www.swift.com/our-solutions/global-financial-messaging/fileact

[48] P. Bogetoft, D. L. Christensen, I. Damgård, M. Geisler, T. Jakobsen, M. Krøigaard, J. D. Nielsen, J. B. Nielsen, K. Nielsen, J. Pagter, M. Schwartzbach, and T. Toft, *Secure Multiparty Computation Goes Live.* ACM, 2009. [Online]. Available: https://dl.acm.org/doi/abs/10.1007/978-3-642-03549-4_20

[49] "Cafeobj." [Online]. Available: https://cafeobj.org/

[50] P. Lindsay, "A tutorial introduction to formal methods," 10 1998. [Online]. Available: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.91.5572&rep=rep1&type=pdf

[51] D. Hazel, P. Strooper, and O. Traynor, "Possum: an animator for the sum specification language," in *Proceedings of Joint 4th International Computer Science Conference and 4th Asia Pacific Software Engineering Conference*, 1997, pp. 42–51.

[52] Wikipedia contributors, "Stride (security) — Wikipedia, the free encyclopedia," 2021, [Online; accessed 25-October-2021]. [Online]. Available: https://en.wikipedia.org/w/index.php?title=STRIDE_(security)&oldid=1048068777

[53] f secure, "Threat analysis swift systems and the swift customer security program," https://www.f-secure.com/content/dam/f-secure/en/business/common/collaterals/f-secure-threat-analysis-swift.pdf.

[54] SWIFT, "Swift qualified certificates," https://www.swift.com/swift-resource/17196/download?language=en.

[55] M. Corkery, "Once again, thieves enter swift financial network and steal," https://www.nytimes.com/2016/05/13/business/dealbook/swift-global-bank-network-attack.html, Nov. 2016.

[56] H. Kılınç and A. Küpçü, "Optimally efficient multi-party fair exchange and fair secure multi-party computation," in *Topics in Cryptology — CT-RSA 2015*, K. Nyberg, Ed. Cham: Springer International Publishing, 2015, pp. 330–349.

[57] R. C. Merkle, "A digital signature based on a conventional encryption function," in *Advances in Cryptology — CRYPTO '87*, C. Pomerance, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1988, pp. 369–378.

[58] S. Goldfeder, J. Bonneau, R. Gennaro, and A. Narayanan, "Escrow protocols for cryptocurrencies: How to buy physical goods using bitcoin," in *Financial Cryptography and Data Security*, A. Kiayias, Ed. Cham: Springer International Publishing, 2017, pp. 321–339.

[59] G. Maxwell, "The first successful zero-knowledge contingent payment," Feb 2016. [Online]. Available: https://bitcoincore.org/en/2016/02/26/zero-knowledge-contingent-payments-announcement/

[60] M. Campanelli, R. Gennaro, S. Goldfeder, and L. Nizzardo, "Zero-knowledge contingent payments revisited: Attacks and payments for services," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '17. New York, NY, USA: Association for Computing Machinery, 2017, p. 229–243. [Online]. Available: https://doi.org/10.1145/3133956.3134060

[61] G. Kolata, "The key vanishes: Scientist outlines unbreakable code," Feb 2001. [Online]. Available: https://www.nytimes.com/2001/02/20/science/the-key-vanishes-scientist-outlines-unbreakable-code.html

[62] X. Wang, X. Wei, J. Ma, G. Li, and J. Zuo, *User Portrait Technology and Its Application Scenario Analysis*. New York, NY, USA: Association for Computing Machinery, 2021, p. 64–69. [Online]. Available: https://doi.org/10.1145/3468920.3468929

[63] H. Varudkar, “Hadoop based collaborative recommendation system,” in *Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies*, ser. ICTCS ’16. New York, NY, USA: Association for Computing Machinery, 2016. [Online]. Available: https://doi.org/10.1145/2905055.2905353

[64] A. Lakshman and P. Malik, “Cassandra: A decentralized structured storage system,” *SIGOPS Oper. Syst. Rev.*, vol. 44, no. 2, p. 35–40, Apr. 2010. [Online]. Available: https://doi.org/10.1145/1773912.1773922

[65] F. Chang, J. Dean, S. Ghemawat, W. C. Hsieh, D. A. Wallach, M. Burrows, T. Chandra, A. Fikes, and R. E. Gruber, “Bigtable: A distributed storage system for structured data,” *ACM Trans. Comput. Syst.*, vol. 26, no. 2, Jun. 2008. [Online]. Available: https://doi.org/10.1145/1365815.1365816

[66] L. Zhou, B. Lu, S. Zhang, and L. Qi, “Data cache optimization model based on hbase and redis,” in *Proceedings of the 3rd International Conference on Data Science and Information Technology*, ser. DSIT 2020. New York, NY, USA: Association for Computing Machinery, 2020, p. 31–35. [Online]. Available: https://doi.org/10.1145/3414274.3414279

[67] J. Cederquist and M. T. Dashti, “An intruder model for verifying liveness in security protocols,” in *Proceedings of the fourth ACM workshop on Formal methods in security*, 2006, pp. 23–32.

[68] J. A. Onieva, J. Zhou, and J. Lopez, “Multiparty nonrepudiation: A survey,” *ACM Computing Surveys (CSUR)*, vol. 41, no. 1, pp. 1–43, 2009.

[69] T. Riivo, “Practical applications of secure multiparty computation,” *Cryptology and Information Security Series*, vol. 13, no. Applications of Secure Multiparty Computation, p. 246–251, 2015. [Online]. Available: https://ebooks.iospress.nl/volumearticle/40036