SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY - INFORMATICS

TECHNICAL UNIVERSITY OF MUNICH

Bachelor's Thesis in Informatics

Design and Analysis of a Tap to Pay Protocol for On-Chain Cryptocurrency Payments

Haokun Zheng

SCHOOL OF COMPUTATION, INFORMATION AND TECHNOLOGY - INFORMATICS

TECHNICAL UNIVERSITY OF MUNICH

Bachelor's Thesis in Informatics

Design and Analysis of a Tap to Pay Protocol for On-Chain Cryptocurrency Payments

Design und Analyse eines Tap to Pay Protokolls für On-Chain-Zahlungen mit Kryptowährungen

Author:	Haokun Zheng
Supervisor:	Prof. Florian Matthes
Advisor:	Burak Öz
Submission Date:	15. September 2023

I confirm that this bachelor's thesis in informatics is my own work and I have documented all sources and material used.

Munich, 15. September 2023 Haokun Zheng

Acknowledgments

First and foremost, I would like to thank my supervisor Prof. Florian Matthes for allowing me to write this thesis at his chair even if it was not directly related to any of the ongoing research projects. Receiving the support about a topic that I had proposed and am personally passionate about has been a great privilege and I am very grateful for the opportunity to work on this thesis. I am furthermore incredibly grateful for the support and guidance of my advisor Burak Öz, who was always available to answer my questions and provide feedback on my work. Especially given the more complicated circumstances due to overlapping stays abroad, Burak was always very supportive, flexible and understanding and all the way to the last minute offered his time and help. Without his supportive and flexible attitude as well as his extensive academic and technical expertise, this thesis would have not been possible.

Abstract

Today's digital payment landscape is shaped by a few players only who are charging significant interchange fees for their processing services, continuously putting merchants and general commerce activity under pressure. In recent years, decentralized ledgers and blockchain networks have repeatedly garnered attention as potentially promising infrastructure alternative for financial transactions. On paper, blockchains and cryptocurrencies can indeed offer several benefits over existing traditional payment methods due to their decentralized nature, including significantly lower and amount-independent transaction fees on certain networks, higher degree of transparency and greater democratized access to digital payment methods. However, while scalability developments in the last few years have provided blockchain networks with a competitive transaction volume processing capacity, the complexity of the technology and lack of user-friendly interfaces continue to limit the mainstream adoption of blockchain-based payment solutions. Today, even the most advanced cryptocurrency payment solutions still rely on users scanning QR codes and then processing transactions themselves using their personal devices, which stands in stark contrast to the contactless smart card and mobile based user experience in the fiat world. In order to bridge this gap, this bachelor's thesis first identifies the state of the art and functional & technical requirements for a tap to pay protocol for cryptocurrency transactions before proposing and developing such a novel protocol, which is finally analyzed regarding its performance, security and usability. The findings and proposed protocol of this thesis aim to provide a valuable foundation for further development of blockchain-based payment infrastructure.

Kurzfassung

Der heutige digitale Zahlungsverkehr wird von einigen wenigen Akteuren abgewickelt, welche für die Bearbeitung der Zahlungen erhebliche Gebühren verlangen, aufgrund von welchen Händler und Geschäfte kontinuierlich unter Druck gesetzt werden. In den letzten Jahren haben dezentralisierte Ledger und Blockchain-Netzwerke immer wieder Aufmerksamkeit als potenziell vielversprechende Alternative für Finanztransaktionen gewonnen. In Theorie könnten Kryptowährungen und ihre Netzwerke aufgrund ihrer dezentralen Eigenschaft in der Tat mehrere Vorteile gegenüber herkömmlichen Zahlungsmethoden bieten, darunter deutlich niedrigere und vor allen transaktionshöhenunabhängige Transaktionsgebühren auf bestimmten Netzwerken, ein höheres Maß an Transparenz und ein deutlich verbesserter, demokratisierterer Zugang zum digitalen Zahlungsmethoden. In den letzten Jahren haben zwar kontinuirliche Weiterentwicklungen gewissen Blockchain-Netzwerken bereits eine mittlerweile vergleiche Verarbeitungskapazität für das Transaktionsvolumen verschafft, doch die Komplexität der Technologie und der Mangel an benutzerfreundlichen Anwendungen schränken die allgemeine Akzeptanz von Blockchain-basierten Zahlungslösungen weiterhin stark ein. Selbst die aktuell am benutzerfreundlichsten Kryptowährungs-Zahlungssyteme sind immer noch darauf angewiesen, dass die Nutzer einen QR-Code scannen und die Transaktion dann selbst mit ihren persönlichen Geräten abwickeln, was in einem starken Gegensatz zu der sonst gewohnten Nutzerinteraktion mit kontaktlosen Chipkarten und mobilen Zahlungsmitteln in der traditionallen Zahlungswelt steht. Um diesem Missstand entgegenzuwirken, werden in dieser Bachelorarbeit zunächst der aktuelle Stand der existierenden Zahlungsmethoden und die funktionalen & technischen Anforderungen an ein Tap-to-Pay-Protokoll für Blockchaintransaktionen ermittelt, bevor anschließend ein solches Tap-to-Pay-Protokoll konzipiert und entwickelt wird, welches schließlich hinsichtlich dessen Leistung, Sicherheit und allgemeinen Praxisanwendbarkeit analysiert wird. Die Ergebnisse und das vorgeschlagene Protokoll dieser Arbeit sollen dabei als Grundlage für die weitere Entwicklung einer Blockchain-basierten Zahlungsinfrastruktur dienen.

Contents

1 Introduction & Motivation

Today's digital payment industry is defined by only a handful of service providers. Such a high degree of market concentration has resulted in continuous pressure on merchants and commerce activity that rely on competitive and affordable payment processing services. Just Visa and Mastercard, the two largest payment networks in the world, held over 63% of the global transaction volume marketshare in 2022 [1] and in the United States alone processed over 87% of all card transaction volume in 2022 with Visa owning 61% and Mastercard owning 26% of the marketshare respectively [2]. The remaining marketshare is held by further smaller but similar payment networks such as American Express or Discover. In Europe, proportions are even more drastic and Visa and Mastercard processed over 99% of all credit, debit and prepaid card transaction volume in 2022 with 54% belonging to Visa and 45% to Mastercard [3]. All existing payment service providers operate on a similar business model, mainly charging an interchange fee for every transaction processed. While the specific fee structure varies between different payment service providers and locations and is dependent on the specific merchant category and used card type, it is usually composed of a percentage of the transaction amount plus a fixed fee. Visa and Mastercard both publish their interchange fee rates publicly, revealing a generally lower fee level for the European Economic Area (EEA) compared to the US and a strong increase in fees specifically for credit card transactions compared to debit cards, translating into varying and biannually increasing fees ranging from 0.2% + 0.00€ to 1.65% + 0.35€ for debit cards in the EEA all the way to credit card fees of 1.35% + $0.05 to 3.15% + $0.10 in the US [4][5][6][7]. It is worth noting that at no time in the transaction process are the payment network providers such as Visa and Mastercard exposed to any credit risk, which is instead entirely carried by the issuing banks. Given that and the fact, that the underlying technological infrastructure and associated operating costs for such payment networks are not significantly higher than that of any other real-time communication network (in its simplest form e.g. email services), the significant amount of (percentage-based) fees charged result in extremely high profit margins at the expense of merchants and consumers.

While there are several reason for the continuos existence of this duopoly, one fundamental aspect making it difficult for new entrants to enter the market is the inherent high-amount of trust required by merchants and consumers to function as a payment network and service provider. Even if operating the infrastructure has become quite affordable at any scale, in theory opening up possibilities for new entrants with lower fees, gaining necessary amounts of trust as central intermediary in the market has repeatedly proven to be difficult. With this in mind, it is worth examining the recent developments and improvements of decentralized ledger and blockchain networks, which

have been gaining significant traction in the last decade. One key difference and benefit of blockchain infrastructure is their decentralized nature. State changes and transactions cannot be controlled by a single entity, but instead are governed by a transparent set of rules and protocols, which are enforced by a large amount of network validators. This decentralized property removes the need for a trust-based relationship between merchants and payment service providers, since the network design itself is able to guarantee the validity of transactions and the integrity of the network, making modern blockchain networks a compelling alternative to the existing payment networks even at small deployment scale. Given the resulting possible shift of trust from individual apps & service providers to the infrastructure itself, new entrants have a better chance to compete.

Historically, the two earliest and by now largest blockchain networks Bitcoin and Ethereum have been limited in their ability to function as a payment network due to their inherent design and the resulting limitations in transaction throughput and gas fee prices (transaction fee equivalent used as incentives for the decentralized participants to operate a node in the network). While Bitcoin and Ethereum's current typical transactions per second (TPS) throughput of 7 and 15 respectively [8] falls orders of magnitude short of Visanet's claimed maximum 76,000 TPS capcity [9], recent developments of scaling solutions such as for example Bitcoin's Lightning Network [10] and Ethereum's Polygon [11] as well as entirely new Layer 1 chains (fundamentally different base networks) such as Solana [12] have been able to offer significantly lower transaction fees and higher potential transaction throughput comparable with that of current traditional payment processing networks. The Solana blockchain for example reports to have capacity for over 65,000 TPS while currently already processing up to 5,000 TPS on an everyday basis [13] and requires a gas fee of merely 0.000005 SOL (Solana's native token) for a simple transfer transaction in between two parties, which at the time of writing (August 2023) equates to about $0.00001 [14]. From a technical perspective, it therefore becomes clear that the Solana blockchain for example has the capacity to provide a competitive transaction throughput.
Although potentially suitable high-capacity blockchain solutions have only emerged in the last few years and could therefore still be considered somewhat niche, it is noticeable that they have not gained any significant traction in the mainstream payment industry. While a magnitude of regulatory and public perception factors have an impact on the broad acceptance of blockchain networks as payment infrastructure, one of the major technical inhibitants of the mainstream adoption has been the technology's complicated handling and lack of user-friendly interfaces. Blockchain wallets have been notoriously complex to setup and manage given the need to handle public/private keys and seedphrases and existing applications are typically more complicated to use than their centralized counterparts. Even the most advanced cryptocurrency payment solutions today still rely on users scanning QR codes and then processing transactions themselves using their personal devices, resulting in a user experience that still differs strongly from that of the traditional contactless smart card and mobile device powered payment world. Given some blockchain networks' above mentioned far more competitive transaction

fees and additional potential benefits such as a higher degree of processing transparency and no restrictive closed-circle access (e.g. compared to credit card applications), it is therefore worth examining the design of a tap to pay protocol for blockchain transactions. Such novel protocol is ideally even compatible with any existing merchant infrastructure and would allow users to pay with cryptocurrency wallets in a similar fashion to how they would pay with a contactless smart card or phone today.

This thesis therefore aims to formally advance the state of user-friendly blockchain payment applications by first identifying the state of the art of the current tap to pay payment systems for fiat and cryptocurrencies alike before then moving on to analyzing general functional & technical requirements for a cryptocurrency-based tap to pay protocol and afterwards subsequently proposing and developing a novel contactless tap to pay protocol utilizing the Solana blockchain, which is lastly analyzed regarding its performance, usability and security with regards to general real-world feasibility.

2 State of the Art and Background

In order to meaningfully contribute to the field of contactless blockchain-based payment systems, it is important to first understand the existing state of the art solutions in the field of contactless payments for both fiat and cryptocurrencies. The following chapter therefore first provides a general introduction into the terms and different technologies used for contactless payments using fiat currencies. In this context, we also examine the underlying standards and specifications that are enabling the contactless payment infrastructure, including NFC and ISO specifications and standards. Finally we analyze and discuss the current state of the art of mainstream-intended blockchain consumer payment solutions and what shortcomings they still have in terms of user experience and usability.

2.1 Existing fiat currency contactless payment landscape and technologies

Today, a variety of different payment methods exist that allow consumers to pay for goods and services in a contactless manner. While contactless payments using physical cards by payment networks such as Visa or Mastercard are a common form of contactless payments, mobile phone based payment methods are also becoming increasingly popular. Noteworthy with mobile phone contactless payments is that phone manufacturers such as Apple or Google through their offered mobile payment services Apple Pay and Google Pay respectively, are acting as yet another revenue-accruing intermediary between the consumer and the payment network, typically increasing the total associated transaction fee for the merchant yet again. Especially latter is the reason why specific merchants may generally support contactless payments, but not necessarily all mobile phone based payment methods such as Apple Pay or Google Pay. Before we however examine the difference among the various tap to pay payment methods and format, we first provide a general overview into the terms and technologies used in the context of contactless payments using fiat currencies and define a for this work suitable payment model and terminology.

2.1.1 Generic payment model

In order to understand the different payment methods and their respective participants, we first formally define a for our use case suitable payment model with all potentially involved parties and communication channels for a typical contactless payment transaction. While a full payment model with accurate depiction of all involved stakeholders

involves a variety of different participants including multiple banks and possibly further intermediaries and service providers dependent on a merchant's specific setup, we will in the following limit our participating parties to the relevant ones for our contactless payment environment and group further participants involved such as the issuing banks together with the payment network/service provider, since their involvement does not have a significant impact on the contactless payment itself. In a simplified model as depicted in Figure 2.1, we therefore define the following participants and their respective roles:

- **Client/Payor:** The person who intends to make a payment/transaction using their contactless payment card or device.
- **Merchant/Payee:** The business or entity that is being paid in the transaction and accepts contactless payments from customers.
- **Payment Network:** The network (e.g., Visa, Mastercard) that connects card issuers, acquiring banks, and merchants. In our model we further extend the payment network to also include the card issuer and acquiring bank as well as potential intermediaries in the interest of simplicity, since their actions do not have a direct impact on the contactless transaction itself.

Furthermore, we define the following terms and devices taking part in the contactless payment transaction:

- **Client payment method:** The device (typically a NFC-capable credit/debit card or mobile phone) through which the client is able to authorize a transaction.
- **Point of Sale (PoS) Card Terminal:** The card terminal device used by the merchant which is connected to the merchants retail checkout system and able to facilitate the communication in between the client's payment method and the payment network. Typically it is the merchant terminal, that initiates a transaction and provides all necessary transaction information such as the transaction amount and the merchant's recipient information in the payment network.

As illustrated in Figure 2.1, the typical contactless payment transaction involves communication in between at least three participants. In general, a standard contactless payment transaction follows the following process:

After the merchant has sent over a pending payment's transaction information to the merchant's PoS terminal, the terminal initiates the transaction and awaits the client's payment method to be presented (through contactless means). After establishing a connection with the client's payment method, the PoS terminal then requests authorization for the given transaction from the client's payment method, which then in turn possibly prompts the client to verify their identity (e.g. through a PIN or biometric authentication method) before authorizing the transaction and sending back the authorization to the PoS terminal. Finally, the PoS terminal then forwards the authorized transaction to

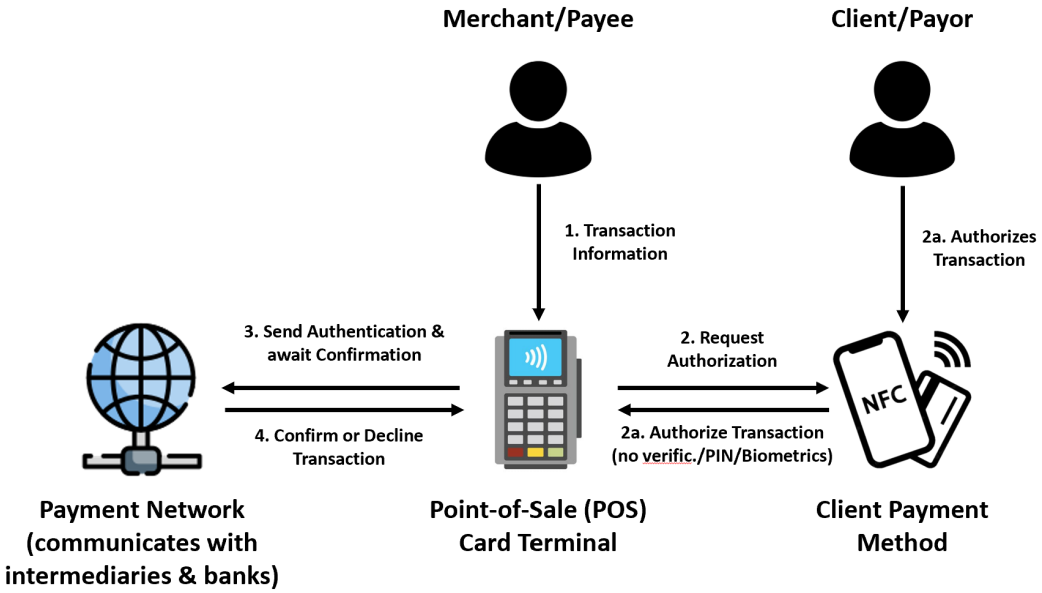


Figure 2.1: Simplified generic contactless payment model with main participants and high-level processing steps.

the payment network, which after processing the transaction internally (in our model simplified) sends back either a confirmation or failure notice to the PoS terminal, which reports this back to the client and merchant's checkout system. Dependent on the specific chosen payment method, it's specific verification configuration and used payment network there might be slight variations to this process, however the general communication flow of a contactless payment transaction can be summarized with the above high-level steps for all contactless payment methods and networks.

2.1.2 Relevant underlying technologies and standards

In order to establish a contactless communication pipeline in between the client's payment method and the merchant's PoS terminal, a variety of different technologies and standards are involved. Before possibly designing a contactless system, it is therefore important to first understand the available technologies and standards and their role in the process. In the following, we will therefore provide a brief introduction into the most relevant technologies and their key characteristics and place them in the overall context of the contactless payment system.

ISO 14443 Standard:

ISO 14443 is an international standard for contactless chip cards and describes general technical characteristics of such smart cards and their corresponding contactless transmission protocols for the communication in between a card and a reader [15]. In a total

of 7 parts (ISO/IEC 14443-1 to ISO/IEC 14443-7), the ISO 14443 standard defines among other things the physical specification of a smart card, the communication protocols including initialization and anti-collision behavior and the underlying technical radio frequency (RF) field specifications to be used to facilitate any communication in between a card and a reader. It is also the ISO 14443 standard that sets the operating frequency for contactless smart cards to 13.56 MHz, which is also the frequency used by NFC devices and tags. The standard is maintained by the International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC) and was first published in 2000 [15]. Today, it serves as the technical foundation for a variety of different contactless smart card applications, including contactless payment cards, electronic identification cards and public transport ticketing cards.

NFC (Near Field Communication):

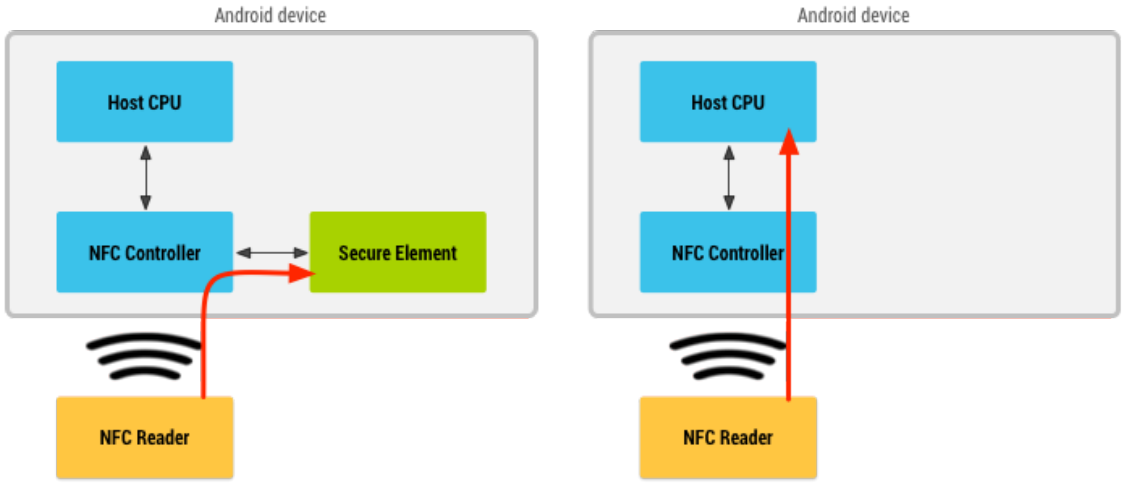
Near Field Communication (NFC) is a set of wireless communication protocols that facilitate contactless data exchange between devices when they are in close proximity, generally within 4 centimeters of each other [16]. The technology and specifications were first introduced in 2002 as a joint development project by Philips and Sony built on top of the existing RFID (Radio Frequency Identification) and the ISO/IEC 14443 standard and today are maintained and updated by the NFC Forum, a non-profit industry association [17]. NFC uses the inductive properties in between two antennas built into NFC-capable devices to sense the presence of tags and devices and facilitate any communication. The operating frequency is 13.56 MHz resulting in a transmission rate of 106, 212 or 424 kbit/s, depending on the communication mode [16] and enabling secure and convenient data transmission between devices, which makes it well-suited for a range of applications, including contactless payment systems. The NFC specifications have since then been further enhanced and extended with international standards such as ISO/IEC 18092 [18] and ISO/IEC 7816 [19] to support a even broader range of applications and use cases. Without going into too much detail, ISO/IEC 18092 for example specifies further communication modes and protocols for NFC devices to communicate with each other, while ISO/IEC 7816 defines communication protocols specifically in between smart cards and their readers, including the APDU (Application Protocol Data Unit) format and transmission protocols, which is also used in our implementation later.

In the context of contactless payments, NFC is used to establish a communication channel in between the client's payment method and the merchant's PoS terminal, which is then used to exchange the necessary transaction information and authorization requests. While NFC supports a variety of different tag types and communication modes, our implementation will be based on the NFC Forum Type 3 Tag standard in reader/writer mode, which is the common tag type used in contactless payment systems today [17].

Android HCE (Host Card Emulation):

While above mentioned standards and specifications define the communication in between actual smart cards and compatible readers, one more important component in order to fully understand the technologies at play during a contactless transaction network is the Host Card Emulation (HCE) technology. In our specific example we will be using Android HCE, which is the HCE implementation for Android devices [20]. Host Card Emulation (HCE) services on modern NFC-enabled phones allows the devices to emulate a smart card through customizable user level software and therefore act as a real contactless card to any external terminal. Before HCE gained popularity, a secure element (SE) hardware component was required to store any necessary card information and facilitate the communication in between a mobile device and a reader without much developer flexibility.

Figure 2.2 from the official Android documentation [20] illustrates the differences in between the traditional secure element based NFC architecture and the newer HCE based architecture. By registering a HCE service and corresponding AID (Application IDs) on the Android phone, the device is able to emulate a smart card and therefore act as a real contactless card to any external terminal once the specific AID is selected by the NFC reader. In the real world, contactless payment methods such as Google Pay and Apple Pay have publicly registered and reserved AIDs that when selected by a PoS terminal at the beginning of the protocol exchange, trigger the corresponding HCE service on the phone to launch the corresponding mobile payment app.



(a) NFC card emulation with a secure element. (b) NFC card emulation through a user application.

Figure 2.2: The two potential Android NFC card emulation architectures compared [20].

Now that we have an overview and briefly defined all major key technologies involved in facilitating and processing a contactless transaction, we can now examine the specific workings of the different contactless payment methods and their respective protocols and standards. This will aid in designing and implementing a comparable tap to

pay protocol for blockchain-based payments later. Here we specifically differentiate in between contactless payments using physical cards and mobile phone based payment methods and will examine the most common payment methods in each category.

2.1.3 Physical cards: EMV card contactless payments

When it comes to contactless payments using physical cards, nearly all physical contactless card implementations use the EMV (Europay, Mastercard and Visa) contactless standard as communication protocol for verification and authorization of payments. In itself, the EMV Contactless standard is part of a larger family of "EMV" standards first introduced in 1994 as a joint project by Europay, Mastercard and Visa to create a global standard for chip-based credit and debit cards. Today it is maintained and updated by EMVCo, a consortium of six major card networks including American Express, Discover, JCB, Mastercard, UnionPay and Visa [21]. Is it worth clarifying that while the EMV standard is maintained by EMVCo, the EMV standard is not a payment network itself or exclusively used by the group members, but rather a general set of specifications and standards that define the technical requirements for chip-based payment cards and terminals. EMVCo furthermore manages all related trademarks and patents related to the contactless protocol specifications, including the commonly seen contactless indicator and symbol shown in Figure 2.3 which indicate the support of contactless transactions on cards and terminals respectively.

(a) EMV contactless indicator for cards.

(b) EMV contactless symbol for PoS Terminals.

Figure 2.3: The official EMV contactless indicator and symbol for PoS terminals indicating contactless card acceptance [21].

The EMV Contactless specifications are built on top of ISO/IEC 14443 and NFC standards and define the technical requirements for contactless payment cards and terminals. According to Visa, payment cards that implement the EMV contactless specifications are capable of making contactless transactions from a range of up to 5 centimeters from a terminal reader and require the cards to be held in proximity for 1-2 seconds after which in about an equal amount of time, most transaction get confirmed [22]. While specific card networks have the ability to modify the EMV Contactless specifications to their specific needs, the resulting contactless protocols all follow a general scheme. For our purposes of simply understanding the overall structure of a concrete contactless protocol implementation, we will therefore examine the specific EMV Contactless kernel specifications used by Mastercard, also referred to as "PayPass". While the EMV Contactless specifications extend over thousands of pages and cover a variety of different

use cases and scenarios, we will in the following mainly refer to van den Breekel et al.'s "EMV in a nutshell" survey of the specifications [23] which provides a comprehensive overview of the EMV Contactless specifications and their implementation in the real world.

In general, a EMV protocol session - no matter whether contactless or chip/magnet-stripe-based - is based on simple public key infrastructure (PKI) and can be broken down into the following 5 steps. To further illustrate, the actual EMV contactless protocol exchange between a card and a terminal for Mastercard's Paypass tap to pay application is depicted in Figure 2.4 with exchanged messages being explained throughout the various steps.

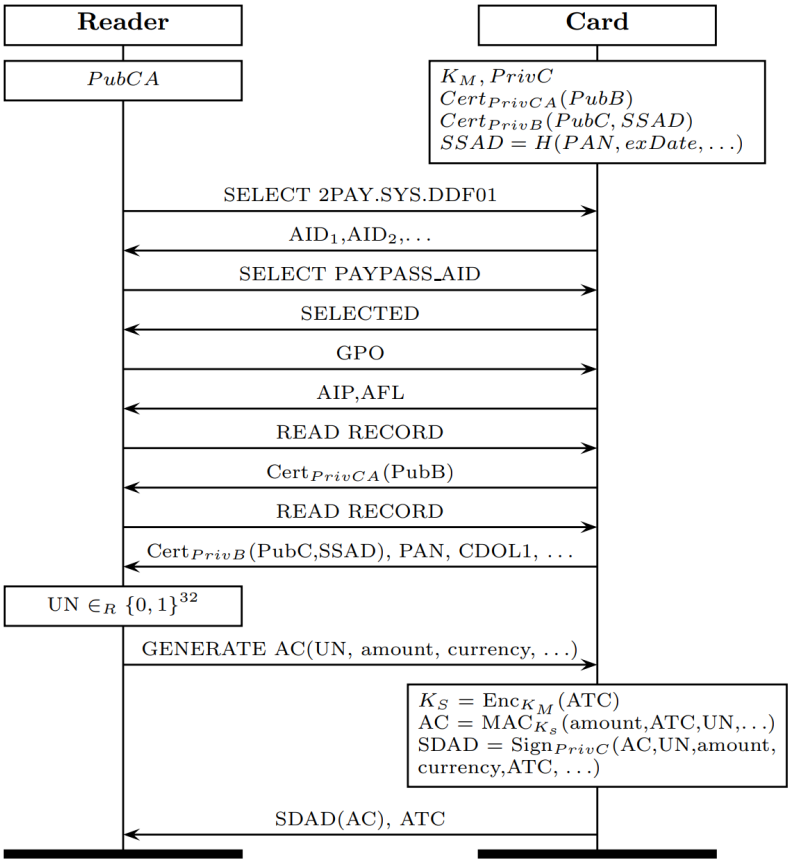


Figure 2.4: Mastercard EMV contactless communication protocol session [23].

1. **Initialization:**
   Before examining the specific steps involved in the initialization of the protocol session, we first need to briefly examine, which values and keys are stored on the

card and terminal respectively. The card generally maintains the following values and keys:

- **Public and private keypair of the card:** A public and private keypair specific to each physical card to authenticate itself and authorize transactions. In Figure 2.4, this is indicated by the variables $Priv_C$ and Pub_C.
- **Symmetric key shared with the card-issuing bank:** In order to preserve backwards compatibility with systems unable to handle asymmetric cryptography, contactless payment cards often still include an additional symmetric key shared in between them and the card-issuing bank. Similarly to the above case, this key can be used to authorize transactions towards the issuing bank. In Figure 2.4, this is indicated by the variable K_M.
- **Signed certificate of the card's public key by belonging card network:** A signed certificate of the card's public key by the issuing card network, which is used by the terminal to verify the authenticity of the card (in the Mastercard protocol indicated with $Cert_{PrivCA}(Pub_B)$ and $Cert_{Priv_B}(Pub_C, SSAD)$).
- **Signed Static Application Data (SSAD) and general configuration set by card network:** Static configuration settings that do not change throughout the lifetime of the card and also similarly covered by a certificate signed by the issuing card network to proof authenticity. Stored configuration values include things such as the *Account number*, *Expiry date of the card* and *Required Verification Methods based on Transaction Amount Thresholds*.
- **Incrementing transaction/interaction counter:** Strictly monotonically increasing counter that is incremented after each transaction or unsuccessful interaction with the card to prevent replay attacks (refer to 3).

The PoS Terminal on the other hand mainly just has to maintain a public key from all payment networks it supports to be able to verify the authentication certificates provided by the various cards it interacts with.

If the client and merchant now intend to initiate a transaction, the client first presents the card to the terminal, which then attempts to establish a connection with the card. After the reader and the client's card have sensed each other's presence, the reader first attempts to learn about the card's supported payment-processing protocols by sending a application selection command to the card, which can be seen in Figure 2.4 as `SELECT 2PAY.SYS.DDF01` command. After receiving the supported Application IDs from the card as reponse, the reader selects the contactless payment application (`SELECT PAYPASS_AID`), which enables it to continue with the following protocol steps.

2. **Card & Data Authentication:**
   After successfully establishing a connection and session for a contactless transaction, the terminal now needs to verify that the card it is communicating with is genuine. In order to do so, it first queries general static card information (SSAD) such as the card number, expiration date as well as maximum transaction amount limits and required cardholder verification measures (stored in

`AIP`, `AFL` in Figure 2.4) and then verifies the certificate of the card's public key ($Cert_{PrivCA}(Pub_B)$). Afterwards, the terminal examines the received SSAD configuration and determines based on the transaction amounts and context, whether further cardholder verification is required or if it can directly proceed to the actual transaction authentication step. It should be noted however, that in the actual real world applications, further safety-related exchanges might occur to prevent a variety of attacks. For example the terminal might request the card to complete a cryptographic challenge within a set time window to rule out relay attacks (see chapter 3). Furthermore, in order to guarantee authenticity of all messages exchanged, the entire communication for all steps below is usually also signed by the card and verified by the terminal every single time, forming an end-to-end encryption communication channel. These details have however been ommitted here as they do not directly aid the general understanding of a typical transaction authentication and authorization process.

3. **Cardholder Verification:**
   If cardholder verification measures are required based on the received SSAD configuration, the terminal now prompts the cardholder to verify themselves using one of the available verification methods. Typically, EMV supports up to three different verification methods, namely *Offline PIN*, *Online PIN* and *Signature*. In the first two cases, the client is prompted to enter a PIN, which is either verified offline by the card itself or online by the card-issuing bank, while in the latter case, the transaction is authorized preemptively and the client is prompted to sign the transaction receipt. In reality however in order to keep the required contact time in between the card and the terminal as short as possible, most contactless transactions today do not require any further cardholder verification and instead opt to simply set a comparatively low maximum transaction limit to limit any potential fraud damages. This is also the case in the given Mastercard protocol session example in Figure 2.4.

4. **Transaction Processing:**
   After both the card authenticity and proper cardholder presence have been verified, the terminal now proceeds to the actual transaction processing step. EMV supports both offline transactions (card holds account balance state that gets updated locally only) as well as online transactions, depending on the specific card configuration and transaction amount. Should the terminal not be able to establish a connection to the payment network, it can also opt to conduct an offline transaction, which generates a transaction certificate that can be sent to the payment network later as proof of a valid transaction. Typically however, the terminals always attempt to conduct an online transaction to limit risk of fraud and to ensure that the cardholder's account balance is sufficient. In order to obtain an authorization for the transaction, the terminal now sends the transaction specifics to the card, which then in turn signs the transaction information and sends it back to the terminal. In the Mastercard example in Figure 2.4, this is indicated by the `GENERATE AC` command where AC stands for "Application Cryptogram". The card updates any

necessary internal states possibly including its account balance and transaction counter and afterwards generates a unique and transaction-counter dependent signed AC object. Together with its new transaction counter value the signed AC object is sent to the terminal, which then in turn forwards the signed AC to the payment network for processing. The specific composition of the AC object is defined by the card network and can include a variety of different information, including the transaction amount, the transaction counter, the cardholder verification method used and the transaction type.

5. **Closing of Connection and potential State Updates:**
   If the transaction was successfully authorized, the terminal now receives a confirmation from the payment network and can report back the status to the client and the merchants own checkout system. Similarly, a declined transaction or failure will be displayed to the client and merchant and if desired a new transaction protocol session can be re-initiated. Depending on the specific implementation, the card and terminal now may optionally also update any internal states using any received responses from the payment network before the protocol session is considered finished and the transaction interaction complete.

Without making any preemptive conclusions and assumptions, it is worth pointing out, that the fundamental underlying transaction protocol session and its reliance on PKI is not necessarily strongly diverging from transaction processes in the blockchain world, possibly indicating a great potential for a rather seamless integration and implementation of blockchain-based contactless payment systems on similar hardware and with a similar user interaction flow.

2.1.4 Mobile phone based payment methods

Mobile phone based contactless payment methods in general are not too different from the above described contactless payment methods using physical cards. While the underlying communication protocols and standards are the same, the main difference is that instead of a physical card, the client's payment method is now a mobile phone with a NFC antenna and the corresponding HCE service. The corresponding services of iOS and Android devices are Apple Pay and Google Pay today, which both work by asking the user to add their credit or debit card information to the corresponding mobile payment app. The mobile payment app then securely stores the card information in its respective online server database. Should a tap to pay interaction be initiated, the mobile payment app now, differently to a typical EMV session, first creates an Google Pay or Apple Pay authentication token that then gets sent to the respective provider's server, which only afterwards uses the client's stored card information on the server to initiate an EMV session with the payment network and process the transaction [24]. Figure 2.5 provides a simplified overview into the adapted transaction model in case of a mobile payment app such as Google Pay or Apple Pay. Important to note is that the specific chain of communication may be dependent on the specific payment service provider, however Figure 2.5 is sufficient for our purposes of simply providing an overview of the

additional intermediary position of Google and Apple. In a sense, the mobile payment

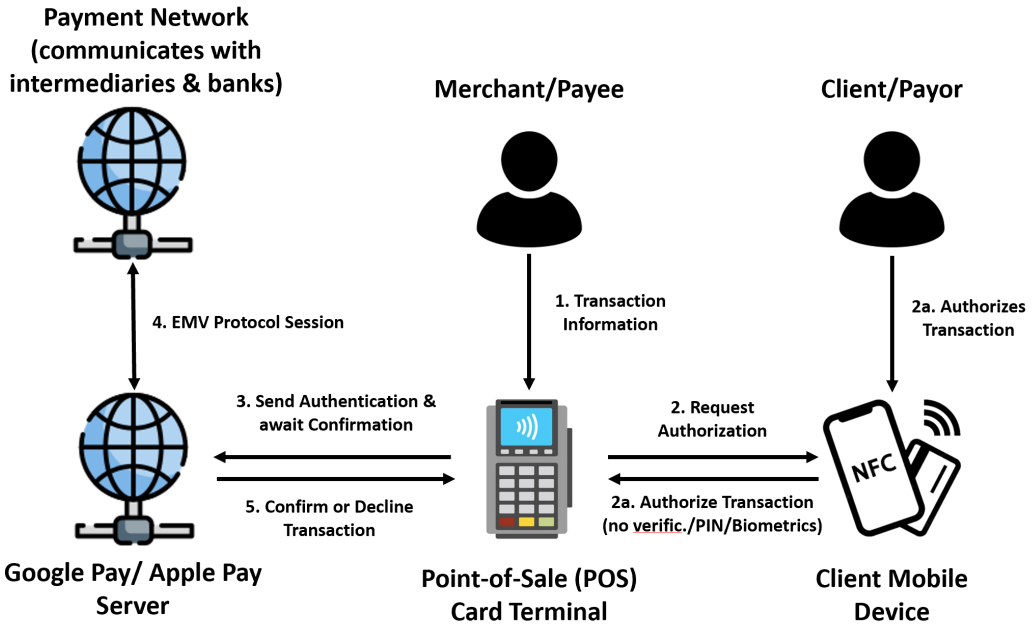


Figure 2.5: Simplified generic payment model of a mobile contactless payment.

service provider therefore acts as an additional intermediary in between the client and the payment network, which is also the reason why some merchants may not support mobile phone based payment methods, since they might not have a contract with the corresponding mobile payment app provider or chose to not want to pay the additional transaction fee for the additional intermediary.

2.2 Cryptocurrencies as payment methods

Blockchains and cryptocurrencies have grown immensely in popularity since they were first proposed in the now infamous Bitcoin whitepaper by Satoshi Nakamoto in 2008 [25]. While the original Bitcoin whitepaper was mainly proposing the creation of a decentralized digital currency without focusing on a specific network capacity or transaction throughput, the underlying decentralized blockchain technology has since been further developed and extended to support a variety of different use cases and applications. Latest with the introduction of Ethereum and its ability to support smart contracts, potential use cases have drastically grown and a variety of applications have been implemented to an extent where today, there exist entire blockchain networks optimized for just one a specific use case. Given our interest in the field of contactless payments, we will in the following therefore focus on the use of cryptocurrencies as a payment currency leveraging their corresponding blockchain networks as payment networks and examine the current state of the art in terms of existing payment methods and protocols.

2.2.1 Blockchain networks and their key characteristics

Before examining the specific payment methods and protocols, it is important to first understand the key characteristics of blockchain networks and their underlying technology. Doing so will later enable us to select the most suitable blockchain network for our use case and to design a protocol that is able to leverage the specific characteristics of the chosen blockchain network.

Public decentralized blockchain networks typically operate under a similar principle: A public blockchain is a distributed network of nodes that maintains a shared ledger of transactions, which is updated and agreed upon by the network participants through a predefined consensus mechanism. That ledger consists of fixed sized blocks of transactions, which are cryptographically linked together in a chain, hence the name blockchain. Dependent on the employed consens mechanism a new block is added to the ledger at a specific intervals, which is typically referred to as the block time. With the addition of so called smart contracts by Ethereum, modern blockchain networks are furthermore able to store data and arbitrary code snippets in addition to simple account balances in the shared ledger, which can then be jointly "executed" by the decentralized network and therefore enable a variety of different applications and use cases. A blockchain network typically has at least one native cryptocurrency, which is used to pay for transactions and incentivize network participants to operate the nodes maintaining the network. Through smartcontracts however, it is also possible to create further tokens, which can also be equally transferred through the network. Especially interesting for our purposes is the fact that these tokens can also represent real world assets, such as for example fiat currencies. Such tokens are then referred to as stablecoins, of which a variety exist on various networks with the largest projects being the US dollar equivalents "USDT" and "USDC" [26][27]. Any individual or entity can "open" an account and identify themselves on the network by generating a cryptographic asymmetric keypair and then use their private key to sign transactions and authorize transfers of funds or other system functions. This keypair identifying a specific account on the blockchain network is also referred to as a wallet and is typically maintained by a wallet application which is running on an user's device. Today, there exist wallet application for almost any device platform, including mobile phones, desktop computers and even dedicated hardware wallets. In order to transfer funds from one wallet to another in the network, the user usually first indicates the amount and recipient's address in their wallet application which then signs the corresponding transaction instructions using the wallet's private key and then broadcast the signed transaction to the network, which then in turn verifies the signature and adds it into a new block which afterwards becomes part of the immutable ledger section.

In order to differentiate in between different blockchain networks and evaluate their suitability for our desired payment network implementation, it is important to understand what characteristics could have what impact on the performance of a potential payment network:

- **Block size:** The block size refers to the maximum amount of data that can be included in a single block of transactions and together with the block time dictate a networks maximum transaction throughput. A larger block size allows for more transactions to be included in each block. This means that a blockchain with a larger block size can accommodate more transactions per block, increasing its transaction throughput to a certain point until, with increasing block size, the corresponding latency for nodes to synchronize regarding new blocks in the network increases to a point the network becomes too prone to forks and orphaned blocks, threatening finality of a potential payment network. In reality however, it is not practicable to directly compare block sizes in between chains to determine their transaction throughput, since different blockchains use different transaction formats and therefore have different transaction sizes. We will therefore also mainly refer to the transaction throughput of a network in terms of transactions per second (TPS) later.

- **Block time:** The block time refers to the average time it takes to create a new block in the blockchain and is typically linked to the chain's underlying consensus mechanism. Block time is usually measured in seconds or minutes. A shorter block time means that new blocks are created more frequently. This results in faster confirmation of transactions, as transactions become part of the immutable section of the chain faster. However, similarly to block sizes, the lower bound for block times is also dictated by the time it takes for nodes to synchronize and reach consensus on the next block, which is why a too short block time can also lead to a higher risk of forked chains and orphaned blocks.

- **Transaction per second (TPS):** The transaction per second metric is a direct measure of the maximum transaction throughput of a blockchain network and is directly linked to the block size and block time. The TPS can be calculated by dividing the block size by the block time. This means that if a block can contain 1000 transactions and the block time is 10 seconds, the network can process 100 transactions per second. In the real world however, an actual TPS metric cannot be statically calculated as transactions have varying complexity and block-space requirements and block times are not constant.

- **Consensus mechanism:** The consensus mechanism is the underlying mechanism that dictates how the network participants reach agreement on the next block (hence, the set of transactions) to be added to the blockchain. The consensus mechanism is therefore also directly linked to the block time and block size and has an immediate effect on the transaction throughput of a network. Typically there are two major types of consensus mechanisms, namely **Proof of Work (PoW)** and **Proof of Stake (PoS)**. In a PoW-based consensus mechanism, the network participants (also referred to as miners) compete to solve a computationally intensive puzzle, which is then used to determine the next block to be added to the blockchain, while in the latter, a node has to "stake" a certain amount of native currency that it risks losing in order to be able to be chosen to create the next block, which

however is no longer compute-intensive. Today, Bitcoin remains the only major blockchain network that still uses a PoW based consensus mechanism, while most other networks have switched to a PoS based consensus mechanism. Further modern blockchains possibly also employ a further development or hybrid of existing consensus mechanisms, such as the Solana blockchain, which we will examine in more detail later. The consensus mechanism is also directly linked to the network's security and decentralization, since it dictates the requirements for a node to be able to participate in the consensus process and therefore also the difficulty to maliciously attack the network.

- **Finality/Confirmation speed:** The finality or confirmation speed of a blockchain network is the time it takes for a block (and its transactions) to be considered final and irreversible. A long time until finality can be problematic for a payment network, since it means that a merchant has to wait a long time until they can be sure that a transaction is final and the funds are theirs.

- **Transaction fees:** Transaction fees (gas fees) are fees necessary to have a transaction be processed and are typically paid by the sender of a transaction to the network participants that are processing the transaction. Transaction fees are usually paid in the native cryptocurrency of the network and are used to incentivize network participants to operate the nodes maintaining the network. Typically transaction fees dependent on the size of the transaction complexity in bytes and the current network congestion and are therefore directly linked to how in demand block space in the current block time interval is. Important to note is that transactions are not guaranteed to be added to the chain and can be rejected by the network if the transaction fee is too low or the network is too congested. The gas fee therefore has an immediate impact on the feasibility of a blockchain as mainstream payment network.

Table 2.1 illustrates the variance in above mentioned characteristics in between some of the most popular blockchain networks today by exemplarily comparing Bitcoin, Ethereum and Solana.

Table 2.1: Differences in blockchain characteristics of Bitcoin, Ethereum and Solana.

Property	Bitcoin	Ethereum	Solana
Whitepaper pub.	2008	2014	2017
Intended Block Time	10min	12s	400ms
2022 Average TPS	5.6	13.5	4,652
Theor. Max TPS	7	Unknown	65,000
Consensus Mech.	PoW	PoS	PoS + Proof of History
Finality: # of block confirmations	6 (60min)	64 (12.8min)	32 (12.8s)

Significantly noticeable is the Solana blockchain's high transaction throughput and low transaction fee compared to the other networks. Without going into too much tech-

nical detail about the specific reasons for the apparent drastic advantage improvement compared to other networks, the later proposed implementation will be based on the Solana network, which is why it is worth characterizing the Solana blockchain a bit further.

2.2.2 The Solana blockchain

The Solana blockchain is a relatively new blockchain network, which was first proposed in 2017 by Anatoly Yakovenko [12] and launched in 2020. It's native token is called SOL and is currently ranked 9th in terms of market capitalization [28]. Similarly to Ethereum, the Solana blockchain is a public blockchain that also supports smart contracts, however unlike other projects, it is not based on any ERC-20 (Ethereum Interface Standard for Tokens) [29] or Solidity (the Ethereum smart contract programming language) [30] standards but instead stands on its own system programs, token and smart contract programming language specification. The Solana blockchain is also based on a Proof of Stake (PoS) consensus mechanism, however unlike other PoS based networks, it uses a hybrid consensus mechanism that combines PoS with a Proof of History (PoH) mechanism, which is used to timestamp transactions before processing by a validator and therefore preserving transaction order information even given an unordered arrvial during transit, enabling a more efficient and faster consensus mechanism[12]. The resulting drastically higher transaction throughput and low transaction fees make it an attractive choice for developers seeking high-performance blockchain infrastructure.

While Solana is not the only high-performance Layer 1 network, it is still the largest and most popular one today based on daily active wallet addresses, and benefits from the largest amount of existing smart contract protocols as well as natively available tokens and stablecoins including Circle's USDC besides the Ethereum ecosystem [31]. It is however noteworthy, that the Solana blockchain has also repeatedly been criticized for a claimed lack of decentralization, which is in part due to higher hardware and staking requirements for validators and therefore a higher barrier to entry for new participants. This sentiment has been further worsened by occasional network outages and downtimes, which could have been a result of its comparative low number of validators required to reach consensus. At the time of writing however (August 2023), the Solana blockchain currently has 1,968 active validators on its mainnet and a superminority of 30 (minimum number of validators required to be able to halt the chain (33% of all staked SOL)) with no history of any malicious activity or centralized attacks [32]. It therefore arguably depends on the application and use case that determines if the Solana blockchain is a suitable choice. For our purposes, the degree of decentralization can be considered sufficient and a reasonable tradeoff, given the chains high transaction throughput and fast time to finality therefore making the Solana network very suitable for our specific intended application.

For the Solana blockchain, similar to the Ethereum Foundation, a Solana Foundation maintains and builds the Solana ecosystem. Additionally and potentially different to other blockchain projects is that there exists a further branch called Solana Labs,

which is structured as a for-profit company that is responsible for the development of standards and applications for the Solana blockchain and its ecosystem. Among the projects and ecosystem-wide standards developed by the Solana Labs team is the Solana Pay framework, which aims to improve the user experience of Solana-based payment applications by offering a variety of features including a QR code based transaction encoding with a standardized payment request format, an API to provide custom and partially signed transaction through an URL and a web interface for online merchants to accept Solana USDC payments [33]. This has led to the development of a variety of further advanced, Solana-based payment applications.

2.2.3 Model of a generic cryptocurrency transaction

In order to better understand the need for a contactless tap to pay protocol, we will in the following first examine the current state of the of existing cryptocurrency payment methods with special respect to their user experience and potential shortcomings. To efficiently do so, we will first briefly adapt our payment model from the generic fiat tap to pay transactions in Figure 2.1 to accurately reflect all generally involved participants during a native cryptocurrency transaction and then examine typical payment scenarios and existing commercial solutions and protocols.

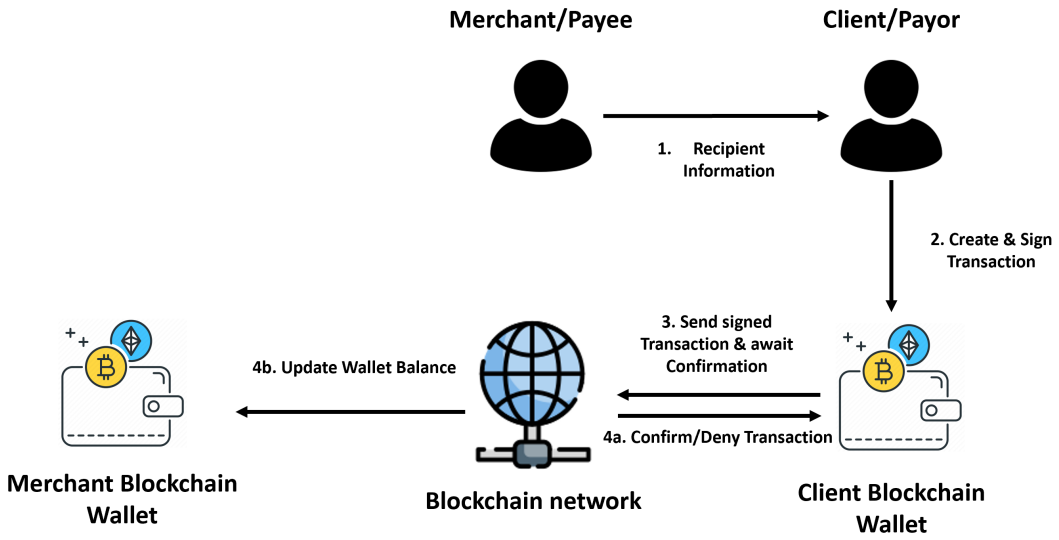


Figure 2.6: Generic cryptocurrency transaction model with main participants and high-level processing steps.

As Figure 2.6 illustrates, a typical crypto transaction does not strongly differ from a traditional fiat transaction in terms of the general participants and processing steps involved. The main participants in a cryptocurrency transaction are the payer and the payee, which are represented by their respective wallets. Both wallets are connected to the blockchain network and are able to send and receive transactions. Worth noting is

that for a regular blockchain transaction, it is the payer that initiates the transaction and is therefore responsible for providing the necessary transaction information. The payee on the other hand is only responsible for providing their address and has no knowledge of an ongoing transaction process until the chain has confirmed the transaction and the payee's wallet has received the funds.

2.2.4 Existing crypto payment methods

Given the higher complexity of a native cryptocurrency transaction compared to a traditional fiat transactions, a variety of different payment methods and protocols have been developed to improve the user experience and enable a more seamless integration of cryptocurrencies as a potential payment method. In the following, we will therefore examine recent improvement proposals and developments from both the academic and commercial world.

Already in the early days of Bitcoin, it became clear that the native Bitcoin transaction format was not suitable for a seamless payment experience. Requiring the user to manually copy-paste a long string of characters representing the recipient's public key and transaction amount into their wallet application was not only cumbersome but also prone to errors, leading to the proposal of a Bitcoin URI scheme [34] as part of the Bitcoin Improvement Proposal (BIP) 21 in 2012, which is a standardized format for encoding Bitcoin transaction information into a URL-String. This development enabled a variety of further consumer-facing developments such as the creation of QR codes that can be scanned by a wallet application to automatically fill in the transaction information. Today, the Bitcoin URI scheme is supported by most major wallet applications and is also used by most major cryptocurrency payment processors to encode transaction information into QR codes for a more seamless payment experience. Other blockchain networks have since brought forward similar URI schemes. Ethereum introduced the standardized format with the Ethereum Improvement Proposal (EIP) 681 [35] and a similar encoding scheme exists for the Solana ecosystem as part of the Solana Pay framework [33].

A variety of different commercial payment methods have since been building on top of the standardized URI and QR code schemes with solutions existing for any ecosystem. Generally we can here differentiate in between three general scenarios and their current state of the art applications:

- **Direct wallet to wallet direct payment scenario:** This scenario describes the simplest direct funds transfer in between two individuals who are both using a native wallet application to conduct a payment transaction. Mobile wallet providers such as Metamask[1] (Ethereum) or Phantom[2] (Solana) typically directly integrate any standardized URI schemes and QR code generation and decoding

[1] https://metamask.io/
[2] https://phantom.app/

functionality into their applications, allowing for swift fund transfer in between two individuals.

- **Client wallet to Mobile PoS Application:** In this scenario, a merchant uses a mobile device with a mobile cryptocurrency PoS application in addition to their existing PoS Terminal to accept crypto payments. A client's mobile crypto wallet application is then able to scan a QR code generated by the merchant's mobile PoS application, which then automatically fills in the transaction information and prepares it for signature. Different to the direct wallet to wallet transfer is that the merchant's mobile PoS application is not just a simple blockchain wallet but usually directly integrated into the merchants inventory and point of sale software or even provides its own corresponding solutions, providing a higher degree of integration and automation. Examples of such applications include cryptocurrency PoS checkout applications like Decaf or Opennode [36][37]. Figure 2.7 shows an example of a typical Decaf mobile PoS checkout screen during a potential client-merchant interaction.

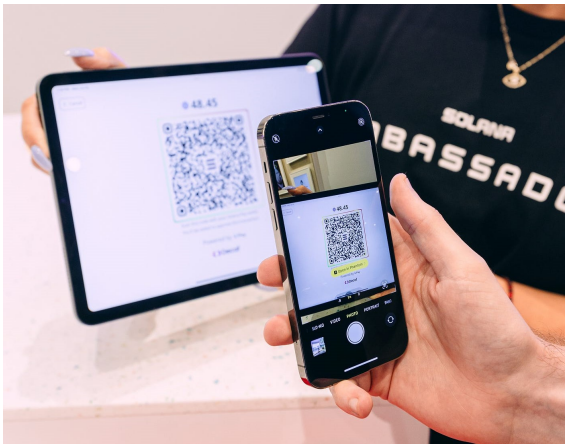


Figure 2.7: Decaf's QR code based payment solution[38].

- **Client wallet to Native PoS Terminal Integration:** In this scenario, a merchant can continue to use their existing PoS terminal hardware which on top of being able to communicate with traditional fiat payment networks, is also able to accept cryptocurrency payments. In order to do so, the merchant has to update their existing PoS terminal with a cryptocurrency payment method provider, which then provides the necessary software to enable the acceptance of cryptocurrency payments. The most popular example of such a solution is BitPay [39], which enables merchants to accept a variety of cryptocurrency payments through their existing PoS terminals through a software update. Figure 2.8 illustrates the integration of BitPay's processing services into an existing PoS terminal.

It is clear that the latter two scenarios are more suitable for a real world application, since they enable a more seamless integration into the existing payment infrastructure

Figure 2.8: BitPay's QR code based PoS terminal integration[40].

and therefore also a more seamless user experience. Especifically the last scenario is desirable for any solutions aiming to be adopted by a larger number of merchants, since it does not require any merchants to obtain new hardware and therefore places a potential blockchain-based payment system in a more directly comparable position towards existing fiat payment networks.

Besides the leading commercial solutions, individual academic works have also examined and proposed end-to-end payment consumer payment applications built on top of the URL/URI encoding standards. In 2014, Eskandari et al. [41] developed a Bitcoin point of sale terminal and compared their solutions to the existing commercial fiat payment systems with regards to user-friendliness, privacy and security. Although the checkout system is simply based on a QR code based payment request, it, according to the authors, went on to be deployed and used in the real world at a cafe for multiple years. Khan et al. [42] in 2019 criticized the security downsides of current solutions requiring client's to conduct transactions with "hot wallets" (crypto wallets that are situated on devices connected to the internet) and proposed a complex multi-step QR-code based checkout process that introduced an intermediary device and step for the client, resulting in a theoretical ability to not using a hot wallet. In 2019, Froehlich et al. [43] furthermore implemented a small scale payment system which was also based on QR codes on the Bitcoin Lightning network and ran an experiment to analyze user behavior and perceived usability of blockchain networks as real-world payment systems. Li et al. [44] in 2020 went a step further and surveyed existing other publications regarding challenges and potential solutions of NFC mobile payment technology applications to facilitate blockchain transactions. Their work however was mainly focused on possibly the comparatively time-consuming QR-code scanning interaction with a NFC tap interaction, which however still requires the user to process the transaction on their mobile device and is therefore only partially compatible with the existing PoS terminal infrastructure.

In their work, Li et al. further determine, that while individual projects and works have made progress in proposing general secure and privacy-preserving payment systems (attack vector resistant & user information remains confidential), many of the papers still fall short on considering or majorly contributing to the user experience and system usability.

2.2.5 Shortcomings of existing cryptocurrency payment methods

After reviewing the existing commercial and academic solutions, it is clear that while there exist a variety of different solutions, none of them are yet able to provide a truly seamless payment experience that is comparable to existing fiat payment systems. This is partially caused by the fact that the existing improvement proposals mainly focus on improving the method in which transaction information is transmitted but still rely on the underlying blockchain network's native transaction process flow in which the client receives the transaction information from the merchant and then has to be the party initiating a transaction. This stands in direct contrast to the traditional fiat payment systems, where the merchant is the party responsible for initiating and processing a transaction and the client only having to authorize the transaction, leading to many usability shortcomings.

When comparing current blockchain payment solutions to exisiting payment networks, one noticeable downside is for example, that all blockchain payment solutions still generally require the client to have their own internet connection and powered device. The client having to be the party initiating the transaction also leads to a variety of other usability issues, including the typical gas fee burden therefore falling onto the client or the merchant not knowing, whether a transaction has been initiated until funds end up arriving in their wallet, which can take up to minutes depending on the chosen underlying blockchain network.

Furthermore from a user experience perspective, all above mentioned solutions fail to provide any meaningful and more convenient alternatives to the scanning of a QR code in comparison to the now ubiquitous tap to pay interaction flows for traditional checkout processes. Depending on the individual mobile wallet configuration, requiring the client to scan a QR code can result in a tedious chain of required interactions starting from unlocking the mobile phone itself over to navigating to and possibly unlocking the mobile wallet application and finally selecting the camera icon and aligning the phone to scan the QR code. This user experience is not competitive with the existing tap to pay payment systems and could be a major barrier to adoption for any blockchain-based payment system even given previously discussed advantages of blockchain-based payment networks.

This observation is also supported by Freohlich et al. [43] who among other things established that the too lengthy QR code scanning interaction being one of the core user experience issues. Furthermore Voskobojnikov et al. [45] also confirm above mentioned

usability shortcomings in a broad user study of mobile cryptocurrency wallets.

Given the upside potential of cryptocurrency networks as mainstream payment network and current state of the art of payment system implementations, it is therefore clear that there is a need for a payment protocol that is able to leverage the advantages of blockchain networks while also providing a user experience that is comparable to existing fiat payment systems. In the following section, we therefore propose our novel tap to pay protocol, which in addition to its contactless NFC-based user interaction, fundamentally rearranges the processing responsibilities of the client and merchant to match the existing fiat payment systems, enabling a truly seamless payment experience while preserving security and compatibility with existing PoS terminal infrastructure.

3 Design of the Tap to Pay Protocol

After establishing the state of the art of payment systems in the previous chapter, this chapter will formally define our methodology and criteria we use to design our tap to pay protocol. In the beginning, we first identify and define functional as well as non-function requirements for a tap to pay protocol for cryptocurrency transactions. These are related to the system design, user experience, software and hardware requirements as well as general assumptions and limitations. Afterwards, we will furthermore define the security requirements for our protocol, which are divided into technical and semantic security requirements. This will be based upon a vast amount of existing literature on NFC tap to pay protocols and their security vulnerabilities. Here it is important to note that while a variety of complex security vulnerabilities and attack vectors exist that modern contactless payment systems are able to mitigate, requiring all of those security properties to also apply to our specific tap to pay protocol implementation lies beyond the scope of this thesis and we instead will be exemplarily implementing a subset of the most critical security characteristics to allow for a basic operation. We however do require that our protocol must generally be designed in a way such that it supports a later addition of all potential corresponding mitigation strategies for all found existing security vulnerabilities to ensure that our protocol maintains real-world implementation relevance and feasibility.

3.1 Design Rationale

After mentioning multiple shortcomings of existing blockchain-based merchant payment systems at the end of chapter 2, we now want to briefly define the overarching rationale and goal of our tap to pay protocol design. While the existing QR code-based payment flow for example is a comparatively time-consuming and tedious process for the client, we do not simply aim to improve the speed of any transaction information transfer through the use of NFC technology as this alone would not enable the client to use cold wallets. Instead, we aim to design novel merchant scenario payment protocol, that is ideally able to simultaneously solve or significantly mitigate all of the previously identified shortcomings of existing blockchain-based payment solutions. In addition to the mitigation of any shortcomings, we also aim to design a tap to pay protocol that is not limited to a specific blockchain network but instead can serve as a general chain-agnostic contactless payment framework for any blockchain and decentralized PKI based payment system. Implementing these requirements and goals not only makes the contribution of this thesis more relevant and generalizable protocol design that can be used as a foundation for future work and tap to pay protocol implementations but furthermore also allows for a better comparability of the protocol with existing solutions

and a more meaningful evaluation of the protocol's performance and security. The following subsection details the specific shortcomings that we aim to address with our protocol design.

3.1.1 Client-side transaction initiation & processing

In nearly all currently existing implementations, the client is the party responsible for initiating the transaction, which leads to a variety of issues:

- **Transaction/Gas fees:** With blockchain transactions, the initiating party of a transaction is typically also the party responsible for paying the transaction fees. While certain newer blockchain networks like Solana allow for a specification of the fee payer when constructing the transaction, making the merchant cover any gas fees in that case requires a further signature by the merchant which in the current way of client-side transaction processing is not possible.

- **Merchant transaction status uncertainty:** Due to the inherent nature of blockchain networks and the existing payment system implementations, the merchant at no time during the transaction knows about the status of the transaction or even if a transaction has been initialized at all since the simple display of a QR code does not allow for any feedback to the merchant. Solely when funds arrive in the merchant's wallet, the merchant can be sure that the transaction has been successfully processed. This however is a major usability issue as it requires the merchant to either trust the client to have initiated the transaction or to wait for the transaction to be confirmed on the blockchain network before handing over the goods or services, which depending on the underlying blockchain network can take up to several minutes and strongly limits the merchant's ability to provide assistance or troubleshoot in case of a failed transaction.

- **Client device requirement:** In order to initiate a transaction, the client is required to carry with them a mobile device with a mobile wallet application installed that is capable of scanning a QR code and initiating a transaction. Requiring a client device to be sufficiently charged might limit the potential use cases in more remote areas and is generally not a requirement for traditional fiat payment systems.

- **Client internet connection requirement:** In addition to requiring a client device, that client is also required to have its own reliable mobile internet connection. While this might be a non-issue for most urban areas in developed countries, this requirement might well represent an issue for less connected areas and stands in contrast to the existing fiat payment infrastructure that often just requires the client to carry around a EMV capable smart card. Specifically regarding the nature of cryptocurrencies, having to initiate a transaction with a powered and internet-connected device that holds the clients private keys also represents a security risk and should ideally be avoided unless absolutely necessary for a system to work.

3.1.2 QR code scanning interaction

The QR code scanning interaction can be a tedious and time-consuming process that is not comparable to the tap to pay interaction flows of existing fiat payment systems. Scanning a QR code can lead to a high number of required inputs and taps on the client's side, starting from the unlocking procedure of the phone all the way to selecting the QR code icon in the mobile wallet application and the fact, that the only existing alternative for the client is to manually enter any transaction information, represents a large shortcoming.

3.2 Functional Requirements

Based on our design rationale, we now define the following functional requirements for our tap to pay protocol:

- **FR1: Contactless Transaction Initialization and Processing.** The protocol must allow for a contactless initialization and processing of a blockchain transaction through use of appropriate contcatless proximity technology in between the merchant's PoS terminal device and a contactless payment device of the client (e.g. a smart card or mobile device). Throughout the entire protocol session, there must not be any physical contact required between the merchant and client devices except for optional verification procedures (e.g. entering a PIN).
- **FR2: Initialization and Device Authentication.** The protocol must allow for an initialization of a transaction by the merchant and be able to optionally conduct an authentication of the client device as authentic and supported payment method.
- **FR3: Client Wallet Ownership/Authorization Verification.** The protocol must support the addition of different methods to verify that the client is authorized to use the wallet that they intend to use for the transaction. Potential verification could for example methods include: *None*, *PIN entry* or *Biometric verification on client device if supported*. Chosen methods must be able to be implemented in a way that does not require any internet connection on the client side.
- **FR4: Transaction Limitations Configuration Support.** The protocol must support the creation and loading of client-dependent configuration of transaction limitations such as maximum transaction amount, maximum transaction amount without verification requirement or maximum number of transactions per time period.
- **FR5: Merchant-side Transaction Processing.** The protocol must allow for a merchant-side processing of the transaction and communication with the underlying blockchain network.
- **FR6: Merchant Transaction Status Feedback.** The protocol must allow for the merchant to receive and maintain status information about the state and success of the transaction from initialization to finalization.

- **FR7: Merchant Transaction Fee Payment.** The protocol must allow for the merchant to have the option to cover any transaction fees for the transaction, even if they are the party receiving funds.

3.3 Non-Functional Requirements

In addition to the functional requirements, we furthermore define the following non-functional requirements for our tap to pay protocol, which we grouped into the subcategories *Usability & Compatibility*, *Performance* and *Miscellaneous*.

3.3.1 Usability & Compatibility Non-Functional Requirements

- **NFR1: Familiarity with existing Infrastructure.** The protocol must be designed in a way that follows similar interaction flows and user experience as existing contactless payment systems to ensure a high degree of familiarity and usability for the end user.
- **NFR2: System State and Status Display.** The protocol must be designed in a way that provides information on its current processing step and allows for a display of the current system state and status to the user and merchant at any time during the transaction.
- **NFR3: Software and Hardware Compatibility.** The protocol must be designed in a way that makes it compatible with existing contactless payment infrastructure PoS terminals and hardware. The merchant should not be required to purchase any additional hardware or software and be able to enable the use of the protocol in a similar fashion to the acceptance of another payment network in their system.
- **NFR4: Use of Common Contactless Standards.** The protocol must be able to be implemented using common contactless standards such as NFC, ISO 14443 and ISO 18092 to ensure a high degree of compatibility with existing contactless payment infrastructure and hardware.

3.3.2 Performance Non-Functional Requirements

- **NFR5: Processing time.** The protocol's processing time for a transaction must be comparable to existing contactless payment infrastructure and during normal operation and expected load on the underlying blockchain networks not exceed 2 seconds of continuos required proximity of the NFC-capable devices.
- **NFR6: Client Hardware & Compute requirements.** The protocol must be designed in a way that all required client-side operations don't require an active internet connection and can also be conducted with a common non-battery powered device such as a NFC-capable smart card with integrated chip for simple cryptographic computations in reasonable time (<2 seconds).

- **NFR7: Scalability.** The protocol must be designed in a way that allows for a high degree of scalability and support for a large number of concurrent transactions. Processing times and required compute should not increase significantly with an increasing number of concurrent transactions.

3.3.3 Miscellaneous Non-Functional Requirements

- **NFR8: System Resilience.** The protocol must be designed in a way such that it will always revert to fail-safe state in case of premature card tear events, connection or power loss.
- **NFR9: Privacy.** The protocol must be designed in a way that does not require the client to disclose any personal information beyond their public key to the merchant or any other third party. Any privacy regulations and requirements applying to traditional payment systems must also be fulfillable by the protocol design.
- **NFR10: Availability.** The protocol must be designed in a way that allows for a high degree of availability and uptime. No unnecessary downtime should be required by the protocol design in between transactions and a continuous repeated use of the protocol should not lead to any system failures or crashes.

3.4 Further Assumptions and Limitations

Besides the functional and non-functional requirements, we furthermore define the following assumptions and limitations in order to provide us with a starting point and framework for the protocol design:

- **A1: Merchant has NFC-capable PoS terminal.** The merchant is assumed to have a NFC-capable terminal device that is able to communicate with the client's NFC-capable device.
- **A2: Merchant has stable internet connection.** The merchant and their PoS terminal are assumed to have a stable internet connection that allows for a communication with the underlying blockchain network. Given the typically stationary nature of the merchant and their hardware unlike the client-side, this assumption is considered reasonable.
- **A3: Client has NFC-capable device.** The client is assumed to have a NFC-capable device that is able to communicate with the merchant's NFC-capable terminal device. This can either be in the form of a NFC-capable smart card with integrated chip or a NFC-capable mobile device.
- **A4: Client and Merchant have a wallet.** The merchant and client are assumed to have wallets and valid wallet addresses that they are able to send and receive funds from the underlying blockchain network. While we do not cover the systems and processes for the merchant to be able to setup and convert received

cryptocurrencies to fiat currency, this is considered a reasonable assumption as the client and merchant are assumed to be to a certain degree familiar with the process of using cryptocurrencies as a payment method.

- **A5: Underlying Blockchain Network is operating normally.** The underlying blockchain network is assumed to be generally operating normally and not be subject to any attacks or malicious behavior for the majority of the time.

3.5 Security Vulnerabilities & Requirements

Although security properties of the protocol technically also fall under the category of non-functional requirements, we decided to define them separately in order to provide a more detailed overview of the security vulnerabilities and requirements that we aim to address with our protocol design.

In the following, we will now sequentially define any applicable known security vulnerabilities and requirements for tap to pay protocols and provide their corresponding mitigation strategies. The following security vulnerabilities and requirements are based on the findings of the EMVCo Contactless Specifications [23] and existing academic literature. Those include Ghosh, Goswami, Kumar, and Majumder [46], Tabet and Ayu [47] and Bellare, Garay, Hauser, et al. [48] who all thoroughly analyzed security vulnerabilities and mitigation strategies of contactless payment systems and Kungpisdan, Srinivasan, and Le [49] who implemented a secure mobile payment protocol. Regarding the given relevant security vulnerabilities and potential attack vectors, we differentiate in between *Tap to pay Protocol related Vulnerabilities* that are a result of the use of contactless communication technologies and *Behavioral/Semantic Vulnerabilities* that are a result of the nature of cryptocurrencies and their underlying blockchain networks as well as potential malicious behavior from one of the participating parties.

3.5.1 Tap to pay Protocol related Vulnerabilities

Unauthorized Usage of Payment Device

- **Definition:** An attacker manages to obtain someone else's payment device or create an authentic copy and uses it to create a valid transaction.

- **Mitigation Strategy 1: Encrypted card contents.** The protocol ideally prevents any information stored on the client-side card or phone to be stored in plaintext and instead requires any sensitive information to be encrypted.

- **Mitigation Strategy 2: Verification Methods.** The protocol must support the addition of different methods to verify that the client is authorized to use the wallet that they intend to use for the transaction. These might include PIN authentication, biometric verification or a combination of both.

- **Damage Limitation Strategy 1: Transaction Limitations.** The protocol must allow the client to configure transaction limitations such as maximum transaction amount, maximum transaction amount without verification requirement or maximum number of transactions per time period such that in case of unauthorized usage, damage can be constrained.
- **Damage Limitation Strategy 2: Wallet Drain Functionality.** In case of loss of a wallet, the client should be able to remotely drain the wallet and transfer the funds to a different wallet address. This could for example be realized as a feature in a accompanying client-side companion mobile application that is used to manage and configure the payment wallet.

Relay Attacks

- **Definition:** An attacker manages to use someone else's card or mobile phone to pay for a transaction by relaying the communication between the merchant PoS terminal to an authentic client device that however is not in actual proximity to the PoS terminal or intended to be authorizing a transaction by the actual owner.
- **Mitigation strategy 1: Timed protocol exchange messages.** The protocol must include time-critical steps that are required to be completed within a certain time frame to rule out signal relaying. With existing EMV payment cards, this is realized by the terminal giving the card a cryptographic challenge for which the card first sends back a processing time estimate that if afterwards broken indicates to the terminal that a relay connection is in place.
- **Mitigation strategy 2: Timeout on terminal side.** The protocol must include a timeout on the terminal side that will abort the transaction if no response is received from the client device within a certain time frame.

Denial of Service Attacks

- **Definition:** An attacker manages to prevent the client device from communicating with the merchant PoS terminal by for example blocking or overtalking NFC signals or the merchant PoS terminal from communicating with the underlying blockchain network.
- **Mitigation strategy 1: Restricted & Authenticated Communication Channels.** The protocol must include limitations on who can communicate with the respective devices on the relevant communication channels and provide possibilities for an optional preceding authentication process.
- **Mitigation strategy 2: Monitor environment for interference.** Due to the close proximity of NFC systems, the merchant and PoS terminal should be able to monitor the terminal's/antenna's surrounding environment for any potential unknown devices or interference and abort the transaction if any are detected.

Man-in-the-Middle (MITM)/Eavesdropping Attacks

- **Definition:** An attacker manages to insert themselves in between the communication between the client device and the merchant PoS terminal and is able to intercept and modify the communication to their advantage. Similarly, Eavesdropping attacks enable the attacker to monitor the communication between the client device and the merchant PoS terminal without modifying it, possibly leading to disclosure of sensitive information.

- **Mitigation strategy 1: Encrypted Communication Channels.** The protocol must support the use of encrypted communication channels between the client device and the merchant PoS terminal to prevent any information from being intercepted and modified without notice.

- **Mitigation strategy 2: Monitor environment for interference.** Due to the close proximity of NFC systems, the merchant and PoS terminal should be able to monitor the terminal's/antenna's surrounding environment for any potential unknown devices aiming to tamper with the communication in between terminal and client and abort the transaction if any are detected.

3.5.2 Behavioral/Semantic Vulnerabilities

Replay Attacks / Attempted Double Spending by malicious Merchant Terminal

- **Definition:** The merchant attempts to resend a signed transaction that he has already sent to the network and been confirmed again or attempts to request a new signed transaction by the client by initially witholding the original transaction to the underlying blockchain network.

- **Mitigation strategy 1: Limited Validity Period for Signed Transactions.** The protocol must ensure that signed transactions are only valid for a limited time period and will be rejected by the underlying blockchain network if not processed within that time frame. Furthermore the chosen blockchain network must rule out a double spend of an identical signed transaction by the same wallet address. This can either be implemented through the consensus mechanism by the underlying blockchain design itself or by including a timestamp in the signed transaction and maintaining a secure list of processed transactions as protocol service providers. It is important to pick a reasonable time frame for the validity period that is long enough to allow for a processing of the transaction by the merchant but short enough to not allow for a malicious merchant to indefinitely block the transaction.

- **Mitigation strategy 2: Unique Transaction Signature.** The protocol must ensure that each transaction is unique, which can be achieved through a nonce or rely on an underlying network, that will maintain a list of already confirmed transactions to identify and reject any duplicate transactions.

Timeout Denial of Service / Malicious Merchant Terminal does not forward signed transaction to network

- **Definition:** The merchant terminal prevents the transaction from being processed by the underlying blockchain network by not forwarding the signed transaction to the network.

- **Mitigation strategy: Limited Validity Period for Signed Transactions.** The protocol must ensure that signed transactions are only valid for a limited time period and will be rejected by the underlying blockchain network if not processed within that time frame. Should a valid transaction expire, the client may have not received the goods or services but the merchant will not have received any funds or be able to do so later. It is important to pick a reasonable time frame for the validity period that is long enough to allow for a processing of the transaction by the merchant but short enough to not allow for a malicious merchant to indefinitely block the transaction.

Wrong/Fraudulent Charge to Wallet

- **Definition:** The merchant attempts to charge the client's wallet with a higher amount than the actual agreed upon transaction amount.

- **Mitigation strategy: Display of Transaction Information on trusted UI Element preceeding Authorization.** The protocol must allow for an intermediary step during client authorization in which the transaction information is displayed on a trusted UI element such as the client's mobile device screen before the client authorizes the transaction.

- **Damage Limitation strategy: Maximum Transaction Amount Limitation.** The protocol must allow the client to configure a maximum transaction amount limitation such that in case of fraudulent charge, damage can be contained.

It has to be noted, that the security issue of a fraudulent charge to the client's wallet is not a security vulnerability of our protocol itself but instead a specific instance of the general principal agent problem [50] and therefore also a general payment system issue in existing fiat contactless payment systems where no trusted UI element is available (e.g. debit/credit cards). This attack vector cannot be completely mitigated through the protocol design itself but instead relies on further higher-level measures such as the implementation of anti-fraud measures by the payment network or operator of the tap to pay solution. These could include but are not limited to an insurance fund, a blacklist program for fraudulent vendors and a restrictive limitation on who can accept payments as merchant in the network.

3.6 Proposed System Architecture

Based on the previously defined functional and non-functional requirements as well as the identified security vulnerabilities and mitigation strategies, we now propose the following general system architecture for our tap to pay protocol:

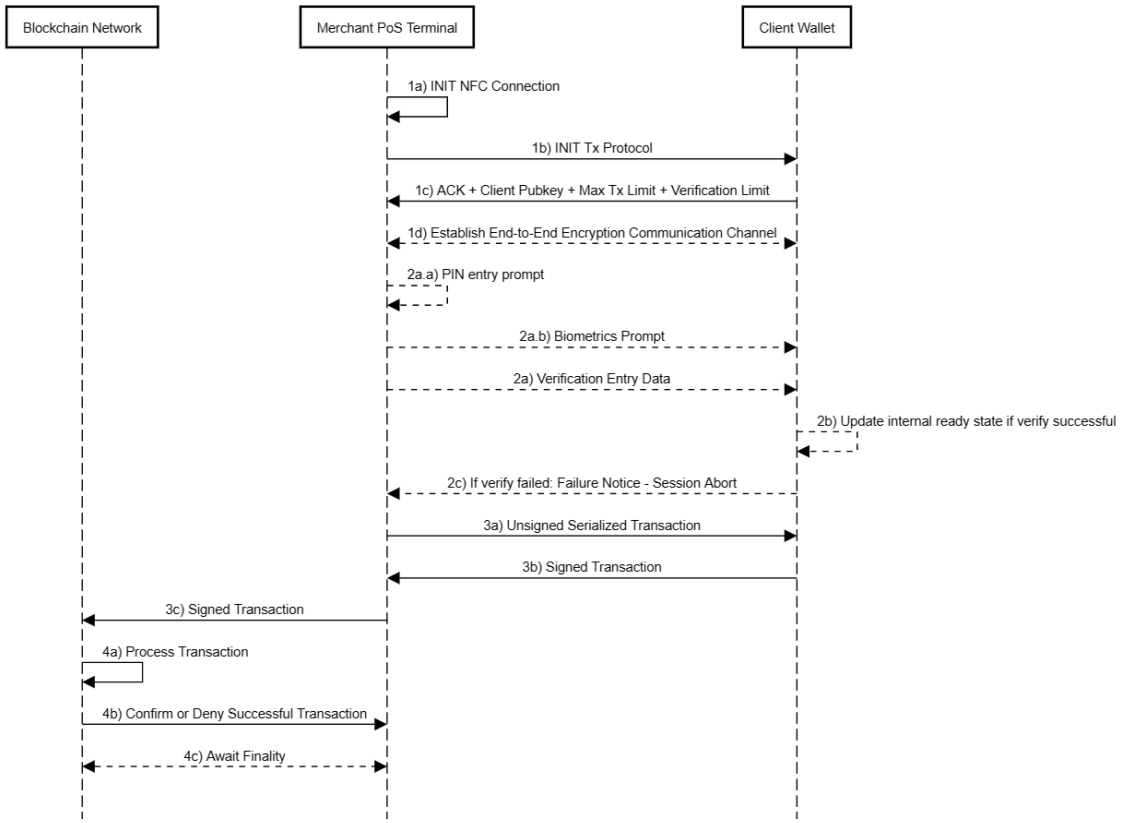


Figure 3.1: System architecture and communication diagram of proposed protocol session.

Upon inspection of Figure 3.1, it can be observed that the system architecture shares some similarities with the existing EMV contactless payment system architecture, which is great for fulfilling the desired compatibility with existing contactless payment systems and hardware as defined in *NFR1*, *NFR3* and *NFR4*. In our proposed architecture we have three interacting parties and components: The ***Client Wallet***, the ***Merchant PoS Terminal*** and the ***Blockchain Network***. The ***Client Wallet*** is the client's payment device that is used to authorize and sign transactions, which could be a chip-based smart card that fulfills the hardware compute requirements defined in *NFR6*. The ***Merchant PoS Terminal*** can be the merchant's existing PoS terminal device that on top of its existing functionality is able to communicate with the client's payment device through NFC and the ***Blockchain Network*** is the underlying chosen blockchain network that is used to process the transaction. The ***Client Wallet*** and ***Merchant PoS Terminal*** are connected

through a NFC communication channel and the ***Merchant PoS Terminal*** and ***Blockchain Network*** are connected through an internet connection, which we assume is given as defined in *A1* and *A2*. The ***Client Wallet*** and ***Blockchain Network*** are not directly connected but instead communicate through the ***Merchant PoS Terminal*** solely through the contactless NFC communication channel, as required in *FR1*.
Based on the above defined involved parties and components, a general protocol session and interaction flow of our tap to pay protocol can be summarized in the following phases & steps:

1. **Initialization and Configuration Loading:**
   a) The merchant initiates the transaction and protocol session through his PoS terminal and awaits a compatible client device to be brought into proximity.
   b) After the client has brought their device into proximity, the merchant PoS terminal sends a protocol initialization message to the client device, signaling the intent to start a protocol session.
   c) The client device receives the initialization message and responds with a protocol initialization response message, which includes initial information about the client device and wallet required for the merchant to be able to prepare the transaction, including the client's public key and transaction and verification limit configuration.
   d) The merchant PoS terminal receives the initialization response message and can optionally establish an encrypted communication channel and verify the authenticity and protocol support of the client device, fulfilling *FR2* and possibly already providing resilience against *Relay*, *Man-in-the-Middle* and *Denial of Service* Attacks. Afterwards, the PoS terminal either aborts the transaction if the clients maximum transaction amount is lower than the transaction amount or proceeds with either *Step 2* or *Step 3* based on the client's verification limit and method settings.
2. **Client Wallet Ownership/Authorization Verification:**
   a) If the client's transaction limit configuration requires a verification method given the current transaction amount, the merchant PoS terminal now starts conducting the verification process and either sends the client device a verification request message which would prompt the client to verify themselves on their client device using biometrics or a message containing a PIN that the client entered on the user terminal.
   b) The client's device which was previously verified as authentic in *Step 1*, updates its internal state now to be willing to sign an incoming transaction as part of the ongoing protocol session in case of a successful client verification and awaits an incoming transaction from the terminal.
   c) In case of a failed client verification, the client wallet sends a verification failure message back to the merchant PoS terminal which could either result in a new verification attempt or an abort of the transaction.

This implementation of a verification process fulfills *FR3* and *FR4* and reinforces compliance with *NFR1*, *NFR2*, *NFR3* and *NFR4*. By conducting a client verification process, the protocol is furthermore able to provide crucial resilience against *Unauthorized Usage*.

3. **Transaction Processing:**

   a) Following either a successful protocol session initialization or client verification process, the merchant PoS terminal now compiles all relevant transaction information into the transaction format of the underlying blockchain and sends this unsigned transaction to the client device for authorization and signature. In the process of assembling the transaction, the client might make a call to the blockchain network to receive time-sensitive information to be included in the transaction, fulfilling *FR5* and implementing the necessary mitigation strategies to defend against *Replay/Double Spending* and *Timeout Denial of Service* attacks. If the merchant choses to pay the gas fees for the transaction, they can for example sign the corresponding part of the transaction before sending it to the client device, fulfilling *FR7*

   b) If the transaction amount is lower than the client's verification requirement limit or if a previous verification process was successful, the client now validates that none of its configured transaction limit policies are violated before signing the transaction and then returning the now valid transaction back to the terminal for broadcast to the network. If the client wallet is a mobile phone, it can also optionally first displays the unsigned transaction information on its own trusted screen to the client and ask the client to confirm their authorization of the transaction, possibly counteracting against Wrong/Fraudulent Charges.

   c) After receiving the signed transaction from the client device, the merchant PoS terminal optionally verifies the integrity and validity of the transaction and its signatures before broadcasting the transaction to the underlying blockchain network and awaiting a confirmation of the transaction.

4. **Transaction Finalization and Closing of Connection:**

   a) After the transaction has been successfully processed by the underlying blockchain network, the merchant PoS terminal receives a confirmation of the transaction and can now finalize the transaction and close the protocol session.

   b) In order to do so, the merchant PoS terminal sends a transaction confirmation message to the client device and both parties can now close the connection. In case of a failed transaction, the merchant PoS terminal can also send a transaction failure message to the client device, which will also result in the closing of the connection and the client can elect to restart a new transaction to try again, representing reversion to a fail-safe state *NFR8*.

c) Depending on the merchant's desired finality configuration, the merchant can either accept the transaction after a custom amount of blocks following the confirmation or await full finality before considering the transaction complete.

Throughout the entire protocol session, the client device and merchant PoS terminal are in close proximity and the client device is not required to have an internet connection, fulfilling *FR1* and *NFR6*. The merchant PoS terminal is furthermore able to receive status information about the transaction at any time during the transaction and can display the current progress of the protocol session to the merchant and client through its integrated display panel, fully satisfying *FR6* and *NFR2*. Given the sequential process flow of the protocol session and the lack of any time consuming processing steps beyond the scope of each individual session, it is reasonable to assume that the protocol will not face any direct scaling or availability/uptime issues, fulfilling *NFR9* and *NFR11*. The proposed protocol architecture furthermore only transmits the minimal required amount of client information to the merchant terminal and does not require the client to disclose any personal information beyond their public key to the merchant or any other third party, furthermore complying with *NFR10*. Lastly, while the specific processing time will have to be observed based on the reference implementation in chapter 4, given the protocols similar amount of compute and communication steps as existing contactless payment systems, we can preemptively assume that the protocol at least has the technical foundation to be quick and not exceed the required processing time of 2 seconds, addressing *NFR5*.

The proposed tap to pay protocol represents a novel approach to contactless payment systems that satisfies all mentioned shortcomings of existing implementations in the Design Rationale section while maintaining a high degree of security and resilience against common security vulnerabilities and attack vectors, making it a truely viable alternative to existing fiat contactless payment systems. The real-world implementability of the protocol is furthermore demonstrated through the reference implementation in chapter 4.

4 Reference Implementation

In order to validate the proposed tap to pay protocol, a reference implementation of the protocol was developed to demonstrate the feasibility of the protocol and to provide a basis for further development of the protocol. The following sections will provide an overview of the hardware and software components implementing the individual tap to pay protocol participants, namely the ***Blockchain Network***, the ***PoS NFC Terminal***, the ***Merchant PoS System*** and the ***Client Wallet***, before illustrating in detail the protocol implementation itself. Given available hardware and software resources, the reference implementation is available through the following GitHub repository: `https://github.com/ZhengHk/cryptoTapToPay`

4.1 Hardware and Software Component Choices

For our reference implementation, we aim to build a tap to pay protocol that is as close to a real-world implementation as possible, which means that we will be using real-world hardware and software components for the implementation wherever possible. This includes the use of a real, live blockchain network and the development of a real client wallet application. Solely for the PoS NFC terminal, we will be developing a simulated counterpart powered with a USB NFC reader device due to difficulties in sourcing a real PoS terminal device with available developer documentation.

4.1.1 The Blockchain Network

While a variety of cryptocurrency networks exist today, the Solana blockchain network was chosen for the reference implementation of the tap to pay protocol. The protocol design itself is blockchain agnostic and can technically be built on top of any blockchain network, however, the Solana network provided a number of advantages over other networks, including:

- **Native Layer 1 Network:** The Solana network is a native Layer 1 blockchain network, which means that it is not built on top of another blockchain network such as Ethereum rollups like Polygon or Optimism. This not only reduces complexity and concerns regarding a too high degree of centralization, but typically also comes with better availability or liquidity of certain tokens and prevents being dependent on the underlying Layer 1 network for transaction finality and security for on- and off-ramping of funds. For the following cases, we will therefore limit our analysis to native Layer 1 blockchain networks, although in theory the protocol could also be built on top of a Layer 2 scaling solutions, which could come with their own benefits.

- **High Transaction Throughput:** Solana's unique Proof of History (PoH) consensus mechanism allows for a high theoretical transaction throughput of up to 65,000 transactions per second (TPS) with the network already processing around 5,000 transactions per second on a daily basis, a significant improvement over other popular :ayer 1 blockchain networks such as Ethereum (15 TPS) or Bitcoin (7 TPS) [12], putting it in contention with existing fiat payment networks.
- **Low Block Time:** One of the major reasons for the network's high TPS is its targeted block time of 400ms, which is significantly faster than any other major blockchain networks today [12]. This low block time not only benefits the potential transaction throughput, which is necessary in order to demonstrate a real-world feasible tap to pay protocol, but also allows for faster transaction finality, which is important for the protocol's security and suitability as real-world payment network. At the time of writing, elementary transfer transactions on the Solana network are typically considered finalized within 1-3 seconds [12] depending on the finality strictness, compared to a few all the way to tens of minutes for other networks.
- **Low Transaction Fees:** Solana's transaction fees are currently set at 0.000005 Sol, which equates to around $0.00001 per transaction at the time of writing [12]. This is significantly lower than the transaction fees of other popular blockchain networks such as Ethereum (around $1.52 per transaction) [51] or Bitcoin (around $1.82 per transaction) [52] and could make the protocol price competetive against traditional payment methods.
- **Native Token Support:** Given Solana's smart contract support, it is currently the only high-speed layer 1 chain that also already has a variety of natively issued tokens. Especially relevant for our implementation are the availability of native stablecoins such as USDC or USDT, which will most likely be the default currency in an actual real-world application.
- **Ecosystem Size:** While the Solana network is still relatively young, it has already attracted a significant amount of attention and funding, with the network's native token SOL being the 9th largest cryptocurrency by market capitalization at the time of writing [28]. This is important as it ensures a certain degree of network security and stability, which is especially important for a payment network and although debatable, we will assume that the network is sufficiently decentralized with its current 1,968 active validators for the purpose of this thesis. Solana's ecosystem size benefit also means that there is already a variety of existing libraries, wallet applications and other infrastructure available, which can be leveraged for the implementation of the protocol.
- **Feature Set and Built-in Security:** The Solana network itself provides us with a lot of critical features and security mechanisms such as for example the ability to individually determine the fee payer of a transaction or its built in transaction replay resistance which are critical for the protocol's security and real-world feasibility. Given the network's fast block time together with the inclusion of a

recent blockhash in each transaction, valid and signed transactions are typically not valid for longer than 1 minute since the maximum age of the recent blockhash value is set to be 150 blocks. This duration is a fitting time frame for applications as payment network where a client doesn't have to worry about a merchant using a signed transaction later in case of a failed initial transaction. All Validators in addition maintain a list of confirmed transactions from the last 150 blocks and verify, if a signed transaction with a recent enough blockhash has been confirmed before, therefore denying double spend attacks [53].

4.1.2 The PoS NFC Terminal

For the implementation of the PoS NFC terminal, we will be simulating a real PoS terminal device with a USB-powered NFC reader device connected to a computer due to aforementioned difficulties in sourcing a real PoS terminal device with publicly available developer documentation. We however make sure that our reference implementation using the USB-powered NFC reader uses the same underlying technologies and protocols as a real PoS terminal device would use in order to guarantee potential real-world adaptability

The specific USB-powered NFC reader we will be using for our reference implementation is the **ACR122U-A9 NFC Reader**, which is based on the NXP PN532 NFC chip and supports all major NFC protocols such as ISO 14443 A/B, MIFARE, FeliCa and NFC Forum Tag Type 1-4 [54]. The reader can be seen in Figure 4.1. As one of the

Figure 4.1: The ACR122U NFC reader.

most popular and widely used NFC reader device, the ACR122U-A9 NFC Reader is also supported by a variety of existing libraries and software, which we will be using

for our implementation. Especially the internally used NXP PN532 NFC chip is the same chip that is used in a variety of other NFC reader devices, including real PoS terminal devices, which means that our implementation can be easily adapted to other NFC reader devices as well. Since the ACR122U NFC reader however is originally intended for consumer use which entails simple tag read and write functionality, we have for our implementation scrapped the original firmware and drivers and reverted to interfacing with the NFC reader through Microsoft's generic WinUSB driver. In order to communicate with the NFC reader, we therefore additionally implemented a custom python program that leverages the `nfcpy` library to establish a secure connection with the NFC reader and to send and receive APDU messages to and from the NFC reader.

4.1.3 The Merchant PoS System

While we have until now only represented the merchant side through their PoS terminal device and assumed, that the PoS terminal will receive transaction information from another merchant Point of Sale checkout system, we have in order for our reference implementation to make sense also implemented the higher level merchant PoS system itself to be able to demonstrate a full purchase and transaction scenario in between a merchant and a client. Furthermore, this segmentation of responsibilities allows us to treat the *PoS NFC Terminal* solely as NFC reader device and to focus on the actual transaction staging implementation in our PoS merchant system with the help of the existing `web3.js` Solana library for `typescript`[55]. For the implementation of the example merchant PoS system, we have chosen to develop a simple web application that is running on a locally deployed `node.js` server. Figure Figure 4.2 shows the web application's example checkout page from which a merchant could initiate a transaction with a client, which if expanded could be further integrated into the merchant's existing accounting, inventory and payment systems.

Figure 4.2: The checkout screen of the custom merchant PoS system before a transaction.

4.1.4 The Client Wallet

For the implementation of the client wallet, we choose to implment the smart card scenario in order to demonstrate the feasibility of the protocol even with the most limited hardware resources. However, since we do not have access to a real smart card device that is unfused and programmable, we will be instead emulating an actual smart card by developing a custom Android application that will be running on an android smartphone in host card emulation (HCE) mode. In HCE mode, a smartphone can accurately emulate a smart card device and from the perspective of the PoS terminal, the smartphone will be indistinguishable from a real smart card device. In addition to the HCE mode, the custom Android application has also been designed in a way to only utilize elementary memory and cryptography functions to guarantee an accurate representation of the smart card's available resources. The custom Android application is built using the Kotlin programming language and the Android SDK and will be executing the client's side of the tap to pay protocol. In our specific implementation, the client wallet application is running on a Huawei Mate 20 Pro smartphone with Android 10, however any android device with Android 4.4 or higher and NFC capability is suitable for the application. The title screen of the application can be seen in Figure 4.3. In order to be able to utilize the NFC functionality of the phone, the wallet app requires the NFC permission to be granted and the phone to be unlocked and in an active state. Furthermore, we have also defined an Application ID (AID) for our client wallet application and registered the AID and the app as a NFC service in the phone's android manifest configuration such that the phone will automatically open the client wallet application from the home screen when it is moved into proximity with the PoS NFC terminal and the corresponding application (through the AID) is selected by the NFC device. The (static) AID for our applicaion is `F987654321` and can in theory be chosen freely as long as it is unique and does not conflict with other known NFC application IDs such as those reserved for the Google Pay App or other installed NFC payment services.

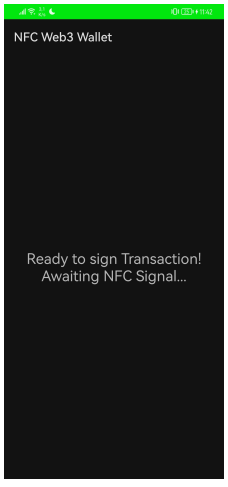


Figure 4.3: The title screen of the custom Android wallet application.

4.2 Simplified System Design

In the following sections, we discuss the specific design and characteristics of our reference implementation of the tap to pay protocol. We begin with the simplifications we made to the general protocol design from chapter 3 and then provide an overview of the system composition and communication in between the individual protocol participants.

4.2.1 Non-restrictive Simplifications

In order to simplify the implementation of the protocol in the interest of remaining within scope of the thesis and to focus on the core protocol functionality, we have made a number of simplifications in our reference implementation with regards to the general protocol proposed in chapter 3. These simplifications however do not prevent the protocol from being extended to support the full protocol functionality and are only made to reduce the complexity of the implementation. The following list provides an overview of the simplifications made in our reference implementation:

- **Payments with Solana's native SOL token:** In our reference implementation, we are creating and processing transactions that are using Solana's native SOL token as the payment currency. Although USDC and other stablecoins are natively available on the Solana Blockchain and most likely the preferred currency in a real-world application, we have chosen to use the SOL native token have to handle a less complex transaction message for demonstration purposes. It is important to note, that the protocol can be extended to support any other token on the Solana network, including stablecoins such as USDC or USDT simply by adapting the specific transaction commands in the Merchant PoS system webapp accordingly.

- **No transaction fee coverage by the merchant:** In our reference implementation, the client remains responsible for the payment of the transaction fee. Similarly to above, this is mainly done to reduce the complexity of the resulting transaction message and the protocol can be easily extended to support the merchant covering the transaction fee by adapting the specific transaction commands in the Merchant PoS system webapp accordingly.

- **No verification of merchant terminal and client wallet authenticity:** In our reference implementation, we are not verifying the authenticity of the merchant PoS system webapp or the client wallet application. In a real-world application, this would be strongly recommended to prevent MITM attacks and to ensure that the client wallet is not communicating with a malicious merchant PoS system. However, our implementation can be easily extended to support the verification of the authenticity of the merchant PoS system webapp and the client wallet application by both carrying a certificate of their public keys that was signed by either a trusted third party or us as protocol and payment service providers.

- **No end-to-end encryption for the entire protocol session:** In our reference implementation, we are not encrypting the entire protocol session in between the client wallet and the merchant PoS system in the interest of readibility of messages. While individual communication channels are potentially secure, such as the NFC communication channel in between the client wallet and the PoS NFC terminal, the communication channel in between the PoS NFC terminal and the merchant PoS system webapp is not encrypted. Here we are strictly speaking sending signed and unsigned transaction details in plain text. This also includes a potential PIN number entered by the client. In a real-world application, and end-to-end encryption channel is highly desired to prevent eavesdropping or MITM attacks, however our simplified implementation does not prevent the protocol from being extended to support end-to-end encryption. This for example could be simply implemented by using the existing public and private keys of the merchant and client wallets for asymmetric encryption of the protocol messages once public keys have been exchanged.

- **No extensive PIN verification procedures and consequences:** In our reference implementation, we only deploy a reduced PIN verification scheme in which a transaction is simply aborted in case of a wrong PIN entry. In real-world applications, a more extensive PIN verification scheme is desired, which could for example include a maximum number of PIN attempts before the client wallet is locked or the wallet's private key is deleted. As with the other simplifications, it is however observable that our implementation could be easily extended to support such a more extensive PIN verification scheme.

- **No timed message check for relay attack protection:** In our reference implementation, we are not timing the duration until a response for a signature request arrives to in order to prevent relay attacks. This however can easily be implemented during any signature activity from the client wallet, e.g. at the beginning of the session when an secure end-to-end encrypted encryption channel is established.

4.2.2 System Design

The reference implementation of the tap to pay protocol consists of three main components, namely the ***Blockchain Network***, the ***PoS Terminal*** and the ***Client Wallet*** which are interacting with each other as illustrated in Figure 4.4.

In the following, we will be providing an overview of the interaction and communication in between the individual components.

4.2.3 Communication in between the Solana Blockchain and Merchant PoS System

The developed merchant PoS webapp leverages the Solana Typescript library to communicate with the Solana blockchain network[55]. Within the library, the webapp in order to create and send transactions is first establishing a connection with a Solana RPC client

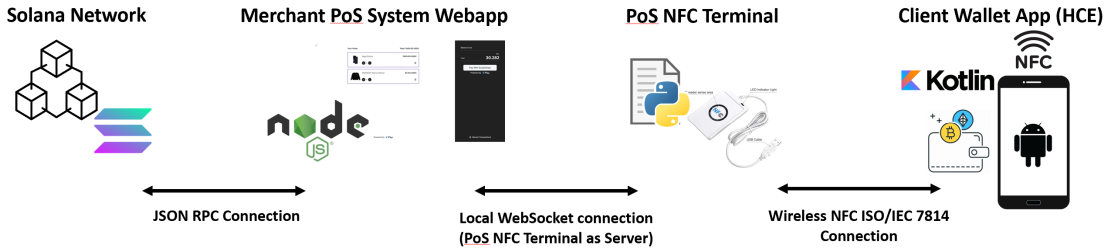


Figure 4.4: The system component overview of our reference implementation

for the corresponding network (`'devnet'` or `'mainnet'`), which then broadcasts a given transaction to the Solana network. The webapp merchant's webapp is also using the Solana Javascript API to query the Solana blockchain for a current blockhash, which is then integrated into the to be created transaction to guarantee transaction replay protection.

4.2.4 Communication in between the Merchant PoS System and PoS NFC Terminal

In our implementation, the merchant PoS system webapp is communicating with the PoS NFC terminal through establishing a `WebSocket` connection in between itself and the custom python program that is running on the computer that is connected to the NFC reader device. The python program in this case is acting as a `WebSocket` server and is listening for incoming messages from the merchant PoS system webapp and establishes a connection with the NFC reader device as soon as the webapp client goes online, putting it in ready state to initiate a protocol session with a client wallet. Information and objects in between the merchant PoS system webapp and the python program are exchanged in JSON-String format and unpacked and interpreted on each end correspondingly.

4.2.5 Communication in between the PoS NFC Terminal and Client Wallet

Communication in between the PoS NFC terminal and the client wallet is happening through the NFC reader device and the android smartphone in HCE mode. Using the `nfcpy` library, the python program is first establishing a connection with the USB-powered NFC reader device and is then waiting for a client wallet to be moved into proximity with the NFC reader device. This is according to NFC and ISO 14443 standards, which results in a distance of roughly 4 centimeters from the reader. Afterwards the python program establishes a type 3 tag connection with the client wallet, which opens a bytearray communication channel in between the two devices. In order to select and address the client wallet application on the smart phone and not any other NFC functionality, the python program initiates an application level communication channel using the ISO 7814 standard for smart card communication and its specified Application

Protocol Data Unit (APDU) packets. In the first APDU that is sent to the client wallet, the python program selects the registered AID of the client wallet application, which the android operating system forwards directly to the client wallet application for further responses and handling. After the client wallet application has received the initial AID selection APDU, it sends the success response code `[0x90, 0x00]` back to the NFC reader, concluding the application selection and communication channel establishment process.

4.3 Tap to Pay Protocol Protocol Session Overview

In the following we will now provide an overview of a tap to pay protocol session within our specific implementation and illustrate specific protocol messages and their contents. To begin, Figure 4.5 shows the general protocol session message flow in between the individual protocol participants, which can be divided into the same overall phases from chapter 3 that we examine further below.

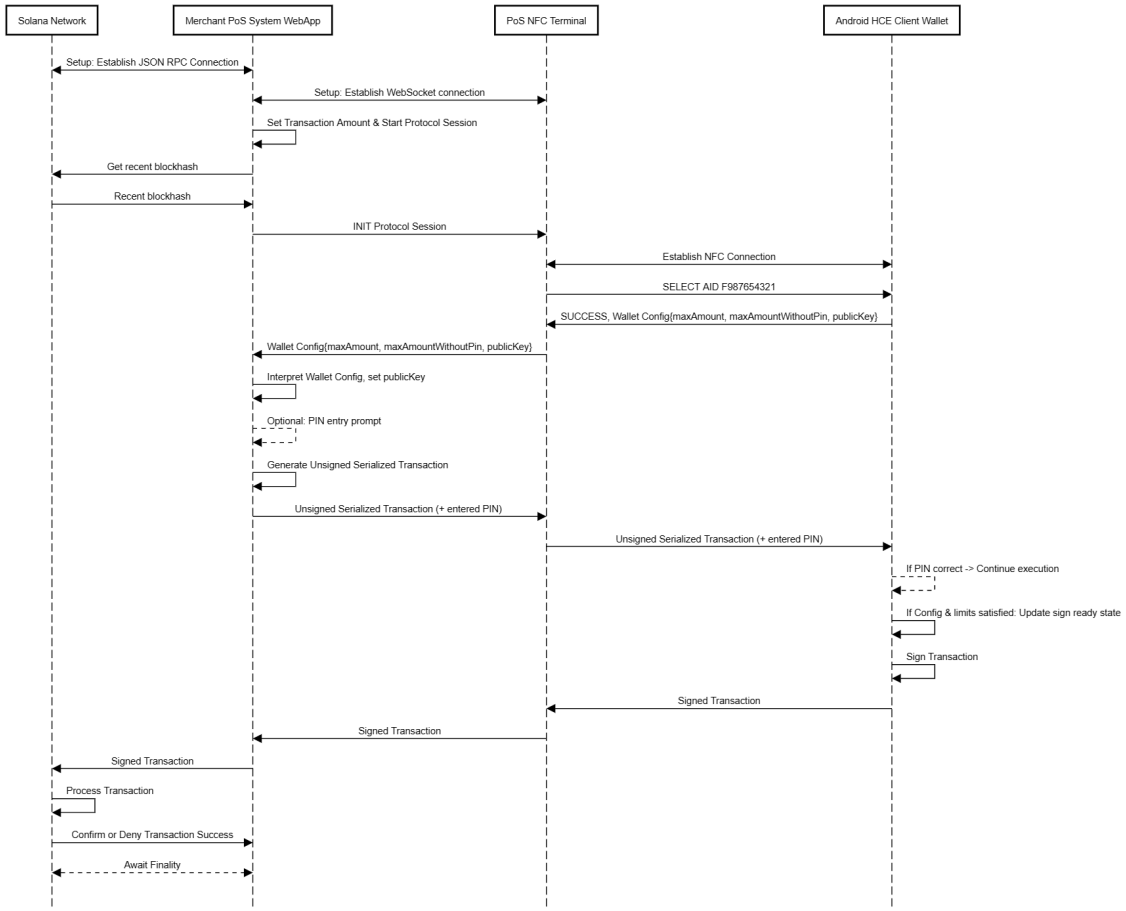


Figure 4.5: Communication diagram of reference implementation protocol session.

4.3.1 Phase 0: Initial States and Variables

Before we begin with the protocol session, we first define the initial states and variables of the individual protocol participants where applicable:

Merchant PoS System

The merchant PoS system webapp initially has a `Transaction Amount` from the current checkout process that it now wants to receive as a payment from a client wallet. Further values and information the webapp is assumed to have is the `Merchant Public Key`. Finally, the webapp has already established a secure `WebSocket` connection with the NFC PoS Terminal python application acting as server.

PoS NFC Terminal

The PoS NFC terminal mainly acts as a communication relay between the merchant PoS system webapp and the client wallet application and therefore does not really manage any major state or variables. Due to the two sided connections with the merchant PoS system webapp and the NFC reader which both produce asynchronous events, the PoS NFC terminal python program deploys a processing thread for each connection and has internal `global variables` to allow both threads to communicate with each other. On the Merchant PoS system facing thread, it acts as `Websocket` server and will have established a connection with the Merchant PoS system webapp before the initiation of a specific protocol session. For implementation simplicity, the PoS NFC terminal python program is also managing a `variable for the AID` of the client wallet application. This however is not necessary and could also be delegated to the Merchant PoS system webapp.

Client Wallet

The client wallet android application represents a fully valid Solana wallet account and therefore manages a `Wallet Public Key` and `Wallet Private Key`. In a real-world scenario, this wallet on the phone or smart card could either have been created by a companion application or imported as an existing account. The `Wallet Public Key` of the the client wallet application is bundled together in a custom `Config` that also contains initial client settings for the maximum transaction amount the wallet can authorize (`maxAmount`) and maximum amount the wallet can authorize without requiring a pin code verification (`maxAmountWithoutPin`). The corresponding object is shown in Listing 4.1.

Listing 4.1: The configuration object stored by the client wallet.

```
object Config {
    val maxAmount = .1
    val maxAmountWithoutPin = .05
    val publicKey = "GA26NywR5aAvs6HswujnfQusBDUSmW6U7rGdrD9GEEM"
    }
```

4.3.2 Phase 1: Session Initialization & Configuration Loading

If the merchant hits the **'Make Contactless Payment'** button on the webapp, the protocol session is formally initiated. Before establishing a NFC connection with the client wallet however, the merchant PoS system webapp first makes a request to the Solana network to verify availability and to retrieve the current `Blockhash` of the network. After this request, the merchant PoS system webapp sends a `Transaction Request` message to the PoS NFC terminal, which prompts the PoS NFC terminal to establish a connection with the USB NFC reader and to wait for a client wallet to be moved into proximity with the NFC reader. The webapp in the meantime displays a spinning loading indicator as can be seen in Figure 4.6 to indicate that the PoS terminal is waiting for a client wallet to be moved into proximity.

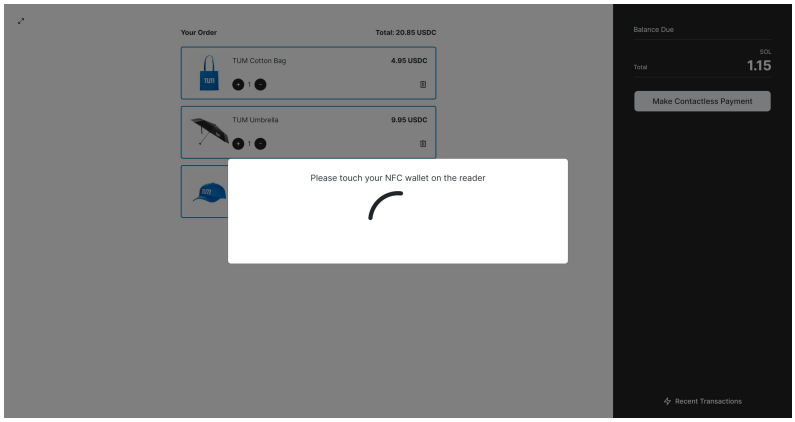


Figure 4.6: The Transaction Process has been initiated and the NFC PoS terminal is awaiting a compatible client device.

After the android phone containing the client wallet app has been moved into proximity of the NFC reader, the NFC reader establishes a connection with the client wallet application and sends an APDU packet containing a command to select the AID `F987654321` to the client phone to establish a secure and direct communication channel with the client wallet application. If all connections are successfully established, the client wallet application will receive the terminal's APDU and respond with the success response code `[0x90, 0x00]` to the NFC reader, which then forwards the response to the merchant PoS system webapp. In order to reduce the amount of required communication and transaction processing time, we additionally also already attach the above mentioned `Config` object to the initial APDU message response that is sent to merchant PoS system webapp. Both success reponse code and the `Config` object can be sent in one message. Afterwards, the merchant PoS system webapp verifies that the current transaction does not exceed the maximum transaction amount set by the client and decides, whether a PIN entry is required or not, proceeding to the next phase accordingly.

4.3.3 Phase 2: PIN Authentication

Should a PIN entry be required by the client wallet, the merchant PoS system webapp will display a PIN prompt and entry field for the client to enter their personal 4-digit PIN. After a 4-digit PIN entry has been made, the merchant PoS system webapp first stores this value internally and proceeds with the assembly and serialization of the transaction. Figure 4.7 shows the corresponding prompt by the webapp.

Figure 4.7: The merchant PoS system webapp is awaiting client PIN input.

4.3.4 Phase 3: Transaction Assembly, Serialization and Signing

Since the merchant PoS system webapp has at this point received the `Config` object from the client, it has knowledge of the client's `Wallet Public Key` (and possibly the just entered verification PIN) and can therefore assemble the transaction accordingly and serialize it, which brings it into its `bytearray` format.

A serialized transaction's structure and components are illustrated in Table 4.1. In our implementation, the merchant PoS system webapp assembles the transaction while leaving the `Signature` field empty for the client to sign (first 64 byte after first byte). The unsigned serialized transaction then gets sent over to the client through the established connection channels (possibly attached with the PIN that was entered), where the client device now verifies the transaction limit compliance (and possibly the attached PIN) before generating a signature over all fields of the transaction beyond the signature area and completing the transaction message. This completed now fully valid transaction is sent back to the merchant PoS system webapp and the NFC connection subsequently closed. Listing 4.2 shows an example of a serialized transaction before and after the client signature has been added. The there used `Public Keys` for the merchant and client wallets are `AYJQ6o82Tr4JfY8gufawH6tMVhMd91tHPwwfdvLoJwSq` and `GA26NywR5aAvs6HswujnfQusBDUSmW6U7rGdrD9GEEM` respectively and the transacted amount is 0.0030282 SOL. While our protocol supports a maximum limit check on the client side before they sign the serialized transaction received from the merchant, we examine and discuss further security and usability requirements and implications of

Table 4.1: Anatomy of a serialized Solana Transaction.

Value	Length in Bytes	Note
Number of Signatures	1	
Array of Signatures	64 * # of Sigs	ed25519 Signature of below section
Number of req. Signatures	1	
Number of read-only Acc.	17	
Number of read-only Acc. without required Signatures	1	
Array of Accs. involved in Transation	32 * # of Accs.	
Recent Blockhash Value	32	SHA-256 Blockhash of Network
Blockchain system instructions	51-1133	Specific system instructions, simple funds transfer requires 51 Bytes

such a system in chapter 3 and chapter 5. The transaction fee is calculated based on the number of to be verified signatures and compute-intensity of the transaction. For an elementary transfer of funds from one account to another such as for our application, the Solana blockchain requires a mere 0.000005 SOL which equates to around $0.00001 at the time of writing.

Listing 4.2: Example of an unsigned and signed serialized Solana transaction.

```
unsignedTx =
[1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,3,3,225,
232,230,199,242,66,243,59,241,42,199,143,157,225,236,210,0,134,154,159,16,
225,79,28,253,159,112,231,87,114,141,189,220,21,246,76,63,240,252,73,143,
211,181,221,37,190,141,93,158,89,80,5,9,25,126,202,57,27,209,106,66,235,
26,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,14,21,
148,219,50,4,28,147,229,75,35,213,201,173,241,244,118,60,40,62,41,168,124,
127,103,64,249,68,219,130,220,204,1,2,2,0,1,12,2,0,0,0,232,52,46,0,0,0,0,0]



signedTx =
[1,14,128,197,42,133,122,252,33,156,231,6,81,145,211,46,112,79,126,251,
220,242,111,17,192,127,201,203,96,189,108,144,14,245,70,76,171,168,235,
114,53,90,13,178,246,195,1,202,217,80,62,77,20,48,175,210,128,149,46,102,
241,134,36,111,6,1,0,1,3,3,225,232,230,199,242,66,243,59,241,42,199,143,
157,225,236,210,0,134,154,159,16,211,225,79,28,253,159,112,231,87,114,141,
```

189,220,21,246,76,63,240,252,73,143,181,221,37,190,141,93,158,89,80,5,9,
25,126,202,57,27,209,106,66,235,26,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,14,21,148,219,50,4,28,147,229,75,35,213,201,173,
241,244,118,60,40,62,41,168,124,127,103,64,249,68,219,130,220,204,1,2,2,0,
1,12,2,0,0,0,232,52,46,0,0,0,0,0]

4.3.5 Phase 4: Transaction Confirmation

After the merchant PoS system webapp has received the completed transaction from the client, it first verifies the transaction signature and then sends the transaction over to the Solana network for processing using the same RPC client connection we previously established with the `web3.js` library. After the transaction has been processed by the Solana network, the merchant PoS system webapp either receives a confirmation with the signature or a failure and error message. In either case, the merchant PoS system webapp displays this as shown in Figure 4.8 including a link to a corresponding Blockexplorer entry in case of a successful transaction.

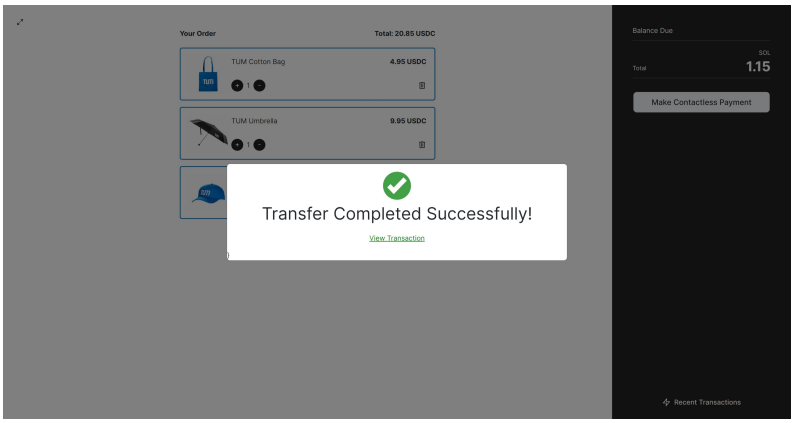


Figure 4.8: The Transaction has been successfully processed by the Solana network.

The protocol session is now concluded and the merchant PoS system webapp is ready to initiate a new protocol session.

5 Analysis

After we have built a reference implementation of the protocol, we will now briefly analyze the protocol with regards to its performance, security and general real-world feasibility in this chapter.

5.1 Performance

The performance of the protocol is one of the key factors for its usability and real-world feasibility. Generally, the performance of the protocol can be characterized in two ways: the duration of the entire transaction and the required contact duration. In the following, we first analyze the performance of our reference implementation by measuring the individual segment times and afterwards compare the performance of our protocol with existing crypto and fiat solutions where applicable and possible.

5.1.1 Setup of measurements

We measure our reference implementations performance through the use of timestamps in the respective code segments. While the accuracy of the timestamps is limited and possibly diluted through the necessary `print` statements required to output the timestamps, we still deem this method sufficient for our purposes since we are not interested in achieving the most accurate measurements possible but rather in getting a rough estimate of the performance of the protocol. In that sense, any influences on the printing of timestamps can be considered negligible compared to the actual execution times. Furthermore, while we do control the hardware and software environment of our reference implementation, we do not control the network environment and thus cannot guarantee that the network performance is not influenced by external factors. This is especially relevant for blockchain networks such as the Solana Network, which's performance is dependent on the network's current load and the number of validators in the network. Therefore, the below measurements should be considered as rough estimates of the performance of the protocol and not as exact measurements or even performance guarantees. Finally, we note that the measurements are based on the performance of our reference implementation which includes minor simplifications as mentioned in chapter 4 with regards to the actually specified protocol from chapter 3. After review of the simplifactions made, we deem it reasonable to assume, that should all simplifications be implemented, the additionally added features will not be adding significant processing times to the protocol session, therefore making our measured performance still valid as order of magnitude estimate.

We define the following individual segments of the protocol for our performance measurements:

- **Entire transaction duration:** The duration of the entire transaction from the moment the client moves their device into the proximity of the reader to the moment the merchant receives the confirmation of the transaction by the Solana network.

- **Required contact duration:** The duration in which the client has to keep their device in the proximity of the reader. This segment ends in the instance, the client has transmitted a valid and signed transaction to the network. It is worth mentioning, that this time segment is independent of the chosen underlying blockchain network and therefore especially relevant for the general usability of our protocol.

To create more reproducable measurements, we disable any PIN-entry requirements for the client's device and assume that the client wallet is able to authorize all transactions without any user interaction. Furthermore, while the reference implementation itself is configured to use the Solana Devnet, we do conduct all transactions that are part of the measurement series on the Solana **Mainnet** to achieve results that represent the real-world performance of the protocol.

5.1.2 Measured Segment Performance

In order to account for any variance in the network performance, we measure the performance of the protocol for a total of five transactions and calculate the average duration of the individual segments. This results in the in Table 5.1 shown processing times of the protocol segments:

Table 5.1: Measured segment performance of the protocol using the reference implementation.

Segment	Duration	Standard Deviation
Entire transaction duration	2.765s	0.052s
Required contact duration	0.893s	0.0073s

5.1.3 Comparison with existing solutions

Following the measurements of the performance of our protocol, we now compare the performance of our protocol with existing crypto and fiat solutions.

Existing cryptocurrency payment solutions

There exist multiple existing cryptocurrency payment solutions and formats as already discussed in the chapter 2. For the comparison of the performance of our protocol with existing cryptocurrency payment solutions, we will focus on the most popular and widely used cryptocurrency payment solution, the QR code-based payment solution, which is also far faster than any even less developed solution relying on manual public key entry. As already seen, there exist multiple QR code-based cryptocurrency payment solutions for a variety of cryptocurrencies. In the interest of comparability, we however focus on the observable processing times of the Decaf PoS solution [36] as it is considered the current state of the art and most popular payment systems in the Solana ecosystem and additionally also designed to be used in physical stores as PoS terminal solution, albeit requiring an additional device. While Decaf doesn't provide any public information on the performance of their solution, we can still gain a rough estimate of their corresponding processing time by analyzing two of their demonstration videos the company published in April 2023 on X (formerly known as Twitter)[56][57]. In the videos, the client uses their mobile wallet to scan the QR code displayed by the merchant's PoS terminal and then confirms the transaction on their device. Similarly, we now measure the time it takes for the client to scan the QR code and confirm the transaction on their device and compare it with the required contact duration of our protocol. Applying similar measurement methods as for our protocol, we measure the entire time from the moment, the client's device scans the QR code until confirmation of the transaction on the PoS terminal and the specific segment of the time the client has to be engaged for the transaction, mainly the time from the moment the client's device scans the QR code until the moment the client confirms the transaction on their device. Table 5.2 shows the measured results for the Decaf checkout system:

Table 5.2: Measured segment times of the Decaf checkout process.

Segment	**Duration**	**Comparison to our Impl.**
Entire transaction duration Video 1	7.59s	2.75x slower
Entire transaction duration Video 2	8.97s	3.24x slower
Required client interaction duration Video 1	3.84s	4.30x slower
Required client interaction duration Video 2	5.48s	6.14x slower

While these measurements are only based on a single video and thus should be considered with caution, they still provide a rough estimate of the performance of the Decaf checkout process. Furthermore, it is important to note that we have already not included the time it takes for the client to open their wallet app and select the QR code scan function in our measurements, which would further increase the processing time of the Decaf checkout process significantly. Our implementation does not require the client to open their wallet app as the client's device automatically launches the app when the device is moved into the proximity of the reader and initiate the transaction process due to us registering the application and its AID as a NFC service in the android manifest.

Even omitting the time it would take the client to open their wallet app the Decaf checkout process is still significantly slower than our protocol. The entire interaction is roughly 3x as long as our transaction processing duration and even more crucially the required client interaction duration is around 5x as long as the required contact duration of our protocol. It has to be noted, that the Decaf checkout process does include a confirmation step on the client's device, which our protocol does not require. However, even factoring in a 1s delay in our protocol due to a slow client confirming a transaction by clicking their device's screen, the required contact duration of our protocol would still be more than 2x as fast as the required client interaction and overall transaction processing duration of the Decaf checkout process. Ignoring further usability downsides of having to scan a QR code and even given the fact, that the measurements can only provide an order of magnitude estimate, our implementation and underlying protocol proofs to be at least very competitive with existing cryptocurrency payment solutions if not several times faster than the existing state of the art.

5.1.4 Comparison with existing fiat solutions

In order to also evaluate the competetiveness and real-world feasibility of our protocol, we now compare the performance of our protocol with existing fiat solutions. Here we focus on the existing EMV contactless payment processes and exemplarily compare our protocol with the contactless payment process of Visa contactless payments. While Visa does not provide detailed public data regarding their observed average duration of a contactless payment, they do claim a maximum duration of 2s for their contactless payment process and regular transaction processing times of "just a few seconds" assuming regular load of their network [22]. These processing times lie within the same order of magnitude than the observed processing for our reference implementation and thus we can conclude that our protocol is performant enough to represent a compelling and competitive alternative to existing fiat contactless solutions from a performance perspective. Furthermore, while Visa itself might be able to confirm the receipt of a transaction within 10s, the underlying involved banks and payment processors often require multiple days to actually conduct and settle the transaction, which is another big advantage of our protocol as it is able to provide near-instant settlement of transactions.

5.2 Usability

Our proposed protocol represents a new approach to cryptocurrency payments and thus it is important to analyze the usability of the protocol and compare it with existing solutions. We therefore first evaluate any usability improvements or downsides compared to the existing QR code based checkout solutions, such as Decaf [36], before comparing the usability of our protocol with existing fiat contactless payment solutions. Table 5.3 provides a comparison of various usability aspects of our protocol compared to existing cryptocurrency and fiat payment systems.

Table 5.3: Comparison of our protocol with current state of the art cryptocurrency and fiat contactless payment systems with regards to various usability aspects. (✓: Mitigation strategy supported by design, (✓): Limited resilience ✗: Open attack vector)

	Our Protocol	**Decaf QR**	**Visa EMV contactless**
Processing Duration	2-3s	7-9s	2-10s
Funds Settlement Duration	0.4-2s	0.4-2s	1-3 days
No Client Input Interaction Required	✓	✗	✓
Supported Client Hardware Platforms	Card & Phone	Phone	Card & Phone
Offline Client Wallet Support	✓	✗	✓
Compatibility with existing Merchant Hardware	✓	(✓)	✓
Transaction Fees (Sept. 2023)	$0.00001	$0.00001	$0.10–$0.35 + 0.2%–3.15%

5.2.1 Usability comparison with existing cryptocurrency payment solutions

As illustrated in the chapter 2, the current state of the art for cryptocurrency payments in physical stores is considered to be a variation of QR code-based payment solutions. These however come with a variety of issues as illustrated in the Design Rationale Section of chapter 3 that our protocol addresses successfully. Namely, the two major issues of QR code-based payment solutions that we identified are the **1) Client-side transaction initiation & processing** and **2) QR code scanning interaction**. The former causes a whole suite of follow-up issues, including the inability of a merchant to cover gas fees, unknown status of a transaction to the merchant and the client-side requirement for a sufficiently powered device that has its own reliable internet connection while the latter is mainly a strongly inferior user interaction method in terms of duration and effort required by the client compared to existing fiat contactless solutions. Our protocol addresses both of these issues by moving the transaction initiation and processing to the merchant's side and by replacing the QR code scanning interaction with a simple contactless NFC interaction that additionally does not require the client to even have the wallet application opened on their device. This approach further brings a variety of smaller benefits such as an enhanced user privacy in which current transaction data is only known to the merchant and the client due to the required proximity in between devices while a sufficiently large and openly displayed personalized transasction QR code might reveal sensitive information to bystanders. We can therefore conclude that our proposed protocol is able to address the major usability issues of existing cryptocurrency payment solutions and thus represents a significant usability improvement over existing solutions.

5.2.2 Usability comparison with existing fiat tap to pay solutions

When comparing our proposed protocol and reference implementation to existing fiat tap to pay solutions such as EMV contactless or Google/Apple Pay payments, we are mainly interested in the degree of similarity of the user interaction duration and the required effort and interactions for the client. The less difference there is between the user interaction of our protocol and existing fiat tap to pay solutions, the more likely it is that our protocol will be able to achieve a similar level of mainstream adoption as existing fiat tap to pay solutions.
Due to the fact, that our protocol was designed in a way that satisfies all imposed functional and non-functional requirements defined in chapter 3, which in turn were designed and derived with the intention to create an as close as possible user interaction to existing fiat tap to pay solutions, we observe that our protocol indeed achieves an almost identical user interaction flow as common fiat tap to pay solutions.

The respective hardware, software and connectivity requirements for a merchant or client are identical to existing fiat tap to pay solutions and in theory compatible with already available merchant PoS terminals or client mobile devices. Additionally, the protocol only utilizes common and already widely adapted contactless technology standards such as NFC, ISO 14443 and ISO 7816 smart cards. In that sense, the protocol in its form could from a technical perspective be easily integrated into existing PoS terminal software as well as existing mobile payment applications such as Google or Apple Pay, cryptocurrency wallets or implemented as a standalone application. Furthermore, due to the specific design of our protocol, an actual implementation based on smart cards is fully feasible and possible, exactly matching the existing fiat tap to pay solutions.

From the client's perspective, the one-tap authorization and signature step is basically identical to existing fiat tap to pay solutions and the ability for the PoS terminal to continuosly provide and display transaction status displays also falls in line with existing fiat tap to pay solutions and represents a benefit over existing QR code-based cryptocurrency solutions. Deployed safety mechanisms such as a maximum transaction amount, PIN-based verification methods or a maximum number of transactions per time period are also already familiar to users from existing fiat tap to pay solutions, all in all enabling a new user to use our protocol immediately without any additional training or learning effort. Finally, the comparative processing speed of our protocol even while using a blockchain network to process a transactions with their additional benefit of real-time transparent transaction settlement compared to a multi-day bank-internal process is a significant improvement over existing fiat tap to pay solutions. In addition to the drastically lower transaction fees, simply through the absence of percentage-based fees, our protocol thus represents, in parts, a significant usability improvement over existing fiat tap to pay solutions and generally provides an almost identical user experience to existing fiat tap to pay solutions.

5.3 Security

The security of our protocol is of utmost importance as the protocol is designed to be used in physical stores and thus has to be able to withstand a variety of attacks. Table 5.4 provides an overview of the security soundness of our protocol with regards to the aforementioned attack vectors and vulnerabilities in comparison to existing crypto and fiat payment solutions.

Table 5.4: Attack vector and security vulnerability resistance of our protocol compared to current state of the art cryptocurrency and fiat payment systems. (✓: Mitigation strategy supported by design, (✓): Limited resilience ✗: Open attack vector)

	Our protocol	Decaf QR	Visa EMV contactless
Unauthorized usage	(✓)	✓	(✓)
Relay attacks	✓	✓	✓
Denial of Service	✓	✓	✓
MITM/Eavesdropping	✓	✓	✓
Replay Attack/Double Spend	✓	✓	✓
Delay Replay/DoS	✓	✓	✓
Fraudulent Charge to Client	(✓)	✓	✗
Cold Client Wallet	✓	✗	✓
Blockchain-Related Consensus Attack (e.g. 51%/Forks)	(✓)	(✓)	N/A

Based on Table 5.4, it quickly becomes apparent that although our protocol is based on a contactless interaction, it does not fall behind to the QR code based client-side processing cryptocurrency payment method in terms of security with exception of the unauthorized usage or fraudulent charge attack vectors. Due to the inherent nature of our protocol being a contactless and NFC based payment method and us allowing a configuration with certain transaction amounts be authorized without client wallet holder verification, our protocol is in theory vulnerable to unauthorized usage attacks in certain configurations while the QR code system will always prompt the user to confirm the transaction on their device. Furthermore our tap to pay protocol is also vulnerable to fraudulent charge attacks in case of a smart card scenario which does not have its own trusted UI element for the client to be able to verify the transaction amount and instructions. These design decision however were made intentionally to allow for a more seamless user experience and most importantly is in line with the existing fiat tap to pay solution's balance between security and usability.

Users that wish to not be vulnerable to unauthorized usage attacks can simply configure their wallet to require a PIN or biometric verification for all transaction amounts. Similarly, using our tap to pay protocol exclusively with a mobile device with trusted UI element will also enable any client to verify the transaction amount before authorizing

the transaction, which is however not a feature of existing EMV and Google/Apple Pay contactless systems, giving our protocol an (optional) security advantage over existing fiat tap to pay solutions.

Finally, it can be observed that our protocol just like the QR code based payment method is still vulnerable to any underlying blockchain-related consensus attacks such as 51% attacks or forks. These attack vectors however are not specifically payment method design dependent and the exposure to those attacks is dependent on the characteristics of the chosen underlying blockchain network such as degree of decentralization or consensus mechanism. In that sense, our protocol is not more vulnerable to these attacks than any other cryptocurrency payment method and thus we do not consider these attack vectors as a disadvantage of our protocol. These vulnerabilities can rather be considered as a general disadvantage of blockchain-based payment methods, which however can be mitigated to a certain degree by choosing a blockchain network with a high enough degree of decentralization and secure design.

5.4 Limitations & Constraints

With every proposed solution, there come limitations and constraints that have to be considered. In the following, we will discuss the limitations and constraints of our protocol and reference implementation in the context of real-world applications. Here we can differentiate in between **technical** and **non-technical** limitations and constraints.

5.4.1 Technical limitations & constraints

Just like with every other technical system, there exist a few technical limitations and constraints that have to be considered when evaluating the real-world feasibility of our protocol. While we do not consider these limitations and constraints as a major disadvantage of our protocol or critical to its real-world feasibility, it is important to be aware of them when considering a potential real-world application. The following list provides an non-exhaustive overview of the most important technical limitations and constraints of our protocol and reference implementation:

Processing performance

The performance of our protocol is dependent on the performance of the underlying blockchain network. While the Solana network is currently able to process transactions within a few seconds, the user experience might well vary when chosing a slower or differently designed network. It is however important to note, that the Solana blockchain is not the only blockchain network of its kind that provides fast and cheap transactions, otherwise making the feasibility of our protocol dependent on one blockchain network only. While their respective ecosystems are smaller than that of Solana, other blockchain projects such as Sui[58] or Aptos [59] have demonstrated similar processing performance

and scalability. Furthermore, existing Layer 2 solutions can also represent a suitable alternative given a sufficient handling of Layer 2 specific behavior.

Transaction fees

The transaction fees of our protocol are dependent on the transaction fees of the underlying blockchain network and its load during the time of the transaction. Unlike fiat payment systems, which determine their transaction fees independent of the current load and mainly based on desired profitability and market competetiveness, blockchain networks' transaction fees inherently depend on the current load and therefore demand of the system. While the Solana chain and other blockchain networks have relatively stable transaction fees for the majority of the time, specific edge cases scenarios in which blockspace demand is spiking might result in a significant increase in transaction fees.

Security of the Protocol

The security of our protocol is similarly to the performance dependent on the security of the underlying blockchain network. While the Solana network is currently satisfying our security requirements, chosing a less secure or differently designed network might well introduce new security vulnerabilities that require additional mitigation strategies or make the protocol unfeasible.

5.4.2 Non-technical limitations & constraints

There exist a variety of non-technical limiations & constraints that impact the actual real-world applicability of our protocol. Especially the reliance on a public decentralized blockchain network may introduce further non-technical complexity when considering real-world deployment. While this thesis does not aim to provide a comprehensive analysis of the non-technical limitations and constraints of a blockchain-based payment solution, we still provide a brief overview of the most important non-technical limitations and constraints of our protocol and reference implementation.

Legal & Regulatory Constraints

One of the major non-technical constraints of our protocol are the legal and regulatory constraints that have to be considered when deploying the protocol in a real-world setting. Given the fact that blockchain and cryptocurrency regulation is still trailing the development of the technology in many parts of the world, the possibility of an actual real-world deployment has to be considered on a case-by-case basis. Even in cases where the use of blockchain and cryptocurrencies is not explicitly prohibited, the legal and regulatory constraints of using a native cryptocurrency or stablecoin tokens as a payment method have to be considered, as many legislative bodies have so far only classified cryptocurrencies as a financial instrument or asset and not as a currency or even as legal tender. While the work and contributions of this thesis do not incorporate any legal or regulatory frameworks or policy recommendations, we still hope to influence the speed

at which blockchain and cryptocurrency regulation is developed and implemented by demonstrating the technologies potential.

Merchant & Client Adoption

While a novel payment method may offer significant advantages over existing solutions, its competetiveness and attractiveness can still vary strongly dependent on the degree of adoption of the payment method by merchants and clients. A payment method that is not used by the majority of a merchant's clientel may not justify the invested effort and cost required to integrate the payment method into the merchant's PoS terminal. Given blockchain's and cryptocurrency's limited current state of adoption as well as divided public perception, it is therefore not sufficient to simply consider technical characteristics when wanting to conduct a complete evaluation of a payment methods real-world feasibility.

On- & Offramping of Funds

Even in a scenario in which regulatory frameworks allow for our protocol to be deployed in a mainstream fashion, an open question that extends beyond this thesis is the incorporation of on- and offramping of funds into the proposed user interaction flow. Our work does not cover the on- and offramping of funds and assumed for protocol sessions that the client had sufficient funds in the cryptocurrency wallet as well as the merchant being able to eventually offramp received cryptocurrencies into their desired native currency. Thus the question of how to incorporate the on- and offramping of funds into the user interaction flow of our protocol remains open. Depending on the legal and regulatory jurisdiction, this problem might not be trivially as any on- and offramping of funds may also require additional KYC/AML checks and thus further complicate the user interaction flow. Since differenct jurisdictions however have varying requirements the successful on- and offramping of funds, the overall system usability including the on- and offramping of funds can vary strongly dependent on the location in question.

5.5 Future Work & Expandalibity

Due to the limited scope of this thesis, certain aspects on the protocol design and reference implementation were simplified where applicable as described in chapter 3 and chapter 4 respectively. While we required that any deviations from a real-world-deployable protocol and reference implementation such as for example simplified hardware components for the merchant PoS terminal must not impact the protocols general security and usability if fully implemented, we now discuss possible future work and expandability of the protocol and reference implementation.

First, future continuing work could focus on the implementation of a complete version of the protocol without the simplifcations made in subsection 4.2.1. These include

usability enhancing features such as the swtich from SOL to stablecoins like USDC as payment currency or enabling of gas fee coverage by the merchant or security enhancing features including a more sophisticated PIN verification scheme or a certificate-based authenticity verification of client and merchant and the subsequent establishment of a end-to-end encrypted communication channel (e.g. by using the client and merchant's available public and private keys).

There also exist further, not already discussed features that could aid the real world adaptability of the protocol. These include but are not limited to the following:

- **Support for multiple blockchains & currencies:** While our protocol currently only supports payments using one previously defined blockchain network, it could be extended to support payments in multiple cryptocurrencies and chains and allow either the merchant or the client to specify which network they desire at the beginning of the protocol session.
- **Smart contract based anti-fraud whitelisting:** In order to mitigate the risk of fraudulent transactions, the protocol could be extended to support a smart contract based anti-fraud whitelisting system. In such a system, both the merchant and client device wishing to support the protocol would have to register their public keys with the smart contract and possibly provide a deposit or other stake. Should any fraudulent activity be detected and independently verified, the party responsible for the fraud would lose their deposit or stake and get blacklisted from using the protocol, which could be indicated in a list managed by a smart contract. A smart contract can also aid in the integration of a variety of other payment system functionality. These could include the ability to issue refunds or the ability to cancel a transaction before it is confirmed by the network if the smart contract temporarily maintains control of the funds.
- **Two-tap hybrid protocol for higher amounts:** In order to broaden potential use cases of the tap to pay protocol and to provide a handling case for higher transaction amounts, a two-tap hybrid model could be implemented. In this scenario, instead of simply declining a transaction that exceeds the maximum configured amount, the protocol could be extended to then pause the contactless interaction and require the client to authorize the specific paused transaction on their mobile device before a second tap on the reader would complete the transaction.
- **Implementation of customer loyalty programs and promotions:** In order to further incentivize the use of the protocol, it could be extended to support the integration with customer loyalty and membership programs based on the public key of the client's wallet. By combining various components of the blockchain technology stack such as for example utility nfts, a whole suite of custom merchant programs and transaction customization could be implemented, possibly providing an even more extensive blockchain-based payment method suite as alternative to the existing fiat systems.

6 Conclusion

In this bachelor's thesis, a tap to pay protocol for cryptocurrency transactions was designed, implemented and evaluated with the aim to improve the general usability of blockchain-based payment systems. To achieve this goal, this thesis first examined how existing fiat contactless payment systems work before moving on to analyze the current state of the art of blockchain-based consumer payment solutions and their shortcomings. Based on these findings, a tap to pay protocol for cryptocurrency transactions was then designed. In order to demonstrate the feasibility of the proposed protocol, a reference implementation was developed, which was then evaluated according to its performance, usability and security properties with regards to a potential real-world deployment.

Following the understanding of today's existing fiat payment systems and their associated fees, we established the general potential of blockchain networks as competetive alternative to existing payment networks. However, we also identified the current usability of existing blockchain-based payment solutions as a major obstacle for their widespread adoption. While there exist a variety of user experience shortcomings for the comparatively difficult and complex to handle technology, the two major issues relevant to payment systems we identified were 1) the comparatively lengthy interaction process for a client to scan a QR code and 2) client-side processing of transactions, which among other things requires the client to carry a capable device that is connected to the blockchain network.

To address both of these issues, we designed a tap to pay protocol for cryptocurrency transactions that allows a client to initiate a payment by simply tapping a contactless card or mobile device to a merchant's point of sale terminal. Hereby, the protocol is not reliant on a specific underlying blockchain network but instead follows a general set of by us defined functional and non-functional requirements, which dictate a more usable and familiar contactless payment experience. In addition, a survey of potential vulnerabilities and list of security requirements was defined to ensure the security and privacy of the protocol.

After the protocol design, a reference implementation was developed to demonstrate the feasibility of the proposed protocol. The reference implementation was built on top of the Solana blockchain, which was chosen because of its favorable transaction throughput and settlement time characteristics. Together with a typescript webapp deployed through a node.js server, an USB-interfacing NFC reader and a custom developed Android mobile app, the reference implementation was able to successfully demonstrate the entire payment process and its potential integration into the existing

merchant payment infrastructure.

Finally, the reference implementation was evaluated according to its performance, usability and security properties with regards to a potential real-world deployment. The performance evaluation showed that the reference implementation is able to process transactions within just a few seconds, which is faster than other state of the art cryptocurrency solutions and competetive with fiat contactless payment systems. In terms of usability, the reference implementation was able to demonstrate a more familiar and intuitive payment experience for the client that matched the usability of existing fiat contactless payment systems without significantly compromising on the security and privacy of the protocol compared to existing QR code-based blockchain solutions. Following a security analysis and comparison with existing payment systems, we observed that our proposed protocol is compatible with any mitigation strategy that is required to defend against all by us identified vulnerabilities where applicable and therefore represents a generally feasible alternative that does not fall back behind existing payment systems in terms of security.

While we were able to demonstrate that a tap to pay protocol for cryptocurerency transactions is feasible and possible given the current state of the art of blockchain technology, there remain limitations as well as opportunities for future work and expansion of the protocol. Limitations and constraints include but are not limited to technical limitations of the underlying blockchain network as well as non-technical limitations such as the potential lack of a regulatory framework for blockchain-based payment systems or open questions regarding the design of surrounding on- and offramping infrastructure. While a thorough analysis and mitigation of these limitations stretches beyond the scope of this thesis, careful attention was paid during the design of the protocol to ensure that the protocol is compatible with any future developments in these areas. There exist a variety of potential future work and expansion opportunities for the protocol, including the support for multiple blockchains or enablement of further payment feature and anti-fraud functionality through the addition of an intermediary smart contract. The contributions of this thesis therefore represent a first step towards a more usable blockchain-based payment system and aim to be a valuable foundation for further research and development of public decentralized blockchain-based payment infrastructure.

List of Figures

List of Tables

Acronyms

AC Application Cryptogram.

AID Application Identifier.

APDU Application Protocol Data Unit.

BIP Bitcoin Improvement Proposal.

EIP Ethereum Improvement Proposal.

EMV Europay, Mastercard and Visa.

HCE Host Card Emulation.

IEC International Electrotechnical Commission.

ISO International Organization for Standardization.

JSON JavaScript Object Notation.

MITM Man-in-the-Middle.

NFC Near Field Communication.

PIN Personal Identification Number.

PKI Public Key Infrastructure.

PoH Proof of History.

PoS Point of Sale/Proof of Stake.

PoW Proof of Work.

QR code Quick Response Code.

RF Radio Frequency.

RFID Radio Frequency Identification.

RPC Remote Procedure Call.

SSAD Signed Static Application Data.

TPS Transactions per Second.

URI Uniform Resource Identifier.

URL Uniform Resource Locator.

USB Universal Serial Bus.

Bibliography

[1] Nilson Report. "Global Network Card Results Worldwide – 2022". In: *The Nilson Report* 1241 (May 2023), pp. 5–8.

[2] Nilson Report. "Global Network Card Card Brands in the US – 2022". In: *The Nilson Report* 1235 (Feb. 2023), pp. 5–8.

[3] Nilson Report. "Europe's Global Network Cards". In: *The Nilson Report* 1247 (Aug. 2023), pp. 5–6.

[4] Visa. *Intra Visa Europe interchange fees - European Economic Area (EEA)*. PDF document. 2021. URL: https://www.visa.co.uk/dam/VCOM/regional/ve/unitedkingdom/PDF/fees-and-interchange/intra-eea-interchange-october-2021.pdf (visited on 08/10/2023).

[5] Mastercard. *Intra-EEA - Intercountry Interchange Fees*. PDF document. 2015. URL: https://www.mastercard.co.uk/content/dam/public/mastercardcom/eu/gb/Other/Website_Intra_EEA_Fallback_Interchange_Fees_220121.pdf (visited on 08/10/2023).

[6] Visa. *Visa USA Interchange Reimbursement Fees*. PDF document. 2022. URL: https://usa.visa.com/dam/VCOM/download/merchants/visa-usa-interchange-reimbursement-fees.pdf (visited on 08/10/2023).

[7] Mastercard. *Mastercard2023–2024U.S. Region Interchange Programs and Rates*. PDF document. 2022. URL: https://www.mastercard.com/global/en/global-wholesale-travel-transaction-interchange.html (visited on 08/10/2023).

[8] L. M. Bach, B. Mihaljevic, and M. Zagar. "Comparative analysis of blockchain consensus algorithms". In: *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. 2018, pp. 1545–1550. DOI: 10.23919/MIPRO.2018.8400278.

[9] Visa. *Visanet Factsheet*. PDF document. 2021. URL: https://usa.visa.com/content/dam/VCOM/global/about-visa/documents/visanet-factsheet.pdf (visited on 08/12/2023).

[10] J. Poon and T. Dryja. "The bitcoin lightning network: Scalable off-chain instant payments". In: (2016). URL: https://lightning.network/lightning-network-paper.pdf.

[11] M. Bjelic, S. Nailwal, A. Chaudhary, and W. Deng. "POL: One token for all Polygon chains". In: (2021). URL: https://polygon.technology/papers/pol-whitepaper.

[12] A. Yakovenko. "Solana: A new architecture for a high performance blockchain v0. 8.13". In: (2018). URL: https://solana.com/solana-whitepaper.pdf.

[13] Solana Foundation. *Solana Network Performance Report: July 2023*. 2023. URL: https://solana.com/news/network-performance-report-july-2023 (visited on 08/17/2023).

[14] Solana Foundation. *Transaction Fees | Solana Docs*. 2023. URL: https://usa.visa.com/pay-with-visa/contactless-payments/contactless-payments.html (visited on 08/31/2023).

[15] International Organization for Standardization (ISO). *ISO/IEC 14443 Standard: Cards and security devices for personal identification — Contactless proximity objects*. Tech. rep. International Organization for Standardization, 2018. URL: https://www.iso.org/standard/73596.html (visited on 08/17/2023).

[16] V. Coskun, B. Ozdenizci, and K. Ok. "A survey on near field communication (NFC) technology". In: *Wireless personal communications* 71 (2013), pp. 2259–2294.

[17] NFC Forum. *About the NFC Forum*. 2023. URL: https://nfc-forum.org/about/leadership (visited on 08/17/2023).

[18] International Organization for Standardization (ISO). *ISO/IEC 18092: Information technology – Telecommunications and information exchange between systems – Near Field Communication – Interface and Protocol (NFCIP-1)*. Tech. rep. International Organization for Standardization, 2013. URL: https://www.iso.org/standard/56692.html (visited on 08/17/2023).

[19] International Organization for Standardization (ISO). *ISO/IEC 7816 Standard: Identification cards – Integrated circuit cards*. Tech. rep. International Organization for Standardization, 2011. URL: https://www.iso.org/standard/54089.html (visited on 08/17/2023).

[20] Google LLC. *Host-based card emulation overview*. 2022. URL: https://developer.android.com/guide/topics/connectivity/nfc/hce (visited on 08/20/2023).

[21] EMVCo. *Overview of EMVCo*. 2023. URL: https://www.emvco.com/about-us/overview-of-emvco/ (visited on 06/15/2023).

[22] Visa. *Visa contactless payments – Learn how to Tap to Pay | Visa*. 2023. URL: https://usa.visa.com/pay-with-visa/contactless-payments/contactless-payments.html (visited on 06/15/2023).

[23] J. van den Breekel, D. A. Ortiz-Yepes, E. Poll, and J. de Ruiter. "EMV in a nutshell". In: (2016).

[24] Google LLC. *Google Pay API | Google for Developers*. 2023. URL: https://developers.google.com/pay/api (visited on 09/11/2023).

[25] S. Nakamoto. "Bitcoin: A Peer-to-Peer Electronic Cash System". In: (May 2008). URL: http://www.bitcoin.org/bitcoin.pdf.

[26] Tether. *Tether token*. 2023. URL: https://tether.to/en/ (visited on 08/31/2023).

[27] Circle Internet Financial Ltd. *Circle | USDC Payments, Treasury Management & Developer Tools*. 2023. URL: https://www.circle.com/en/ (visited on 08/31/2023).

[28] CoinMarketCap Ltd. *Cryptocurrency Prices, Charts and Market Capitalizations | CoinMarketCap*. 2023. URL: https://coinmarketcap.com/ (visited on 09/02/2023).

[29] V. Buterin and F. Vogelsteller. *ERC-20: Token Standard*. 2019. URL: https://eips.ethereum.org/EIPS/eip-20 (visited on 09/11/2023).

[30] Soliditylang.org. *ERC-20: Token Standard*. 2019. URL: https://docs.soliditylang.org/en/v0.8.21/ (visited on 09/11/2023).

[31] CoinMarketCap Ltd. *Top Solana Ecosystem Tokens by Market Capitalization | CoinMarketCap*. 2023. URL: https://coinmarketcap.com/view/solana-ecosystem/ (visited on 09/11/2023).

[32] Solana Foundation. *Solana | Validators*. 2023. URL: https://solana.com/validators (visited on 09/02/2023).

[33] Solana Labs. *A decentralized, permissionless, and open-source payments protocol | Solana Pay*. 2023. URL: https://solanapay.com/ (visited on 09/02/2023).

[34] Bitcoin.org. *Payment Processing – Bitcoin*. 2023. URL: https://developer.bitcoin.org/devguide/payment_processing.html (visited on 09/02/2023).

[35] Ethereum.org. *ERC-681: URL Format for Transaction Requests*. 2023. URL: https://eips.ethereum.org/EIPS/eip-681 (visited on 09/02/2023).

[36] Decaf Inc. *Decaf – Point of Sale Docs*. 2023. URL: https://www.decaf.so/about/pos (visited on 09/02/2023).

[37] OpenNode Inc. *Bitcoin Payment Processor | OpenNode*. 2023. URL: https://www.opennode.com/ (visited on 09/02/2023).

[38] Decaf Inc. *Crypto payments year in review — Decaf*. 2023. URL: https://medium.com/@Decafpago/decaf-web3-year-in-review-5d72d02b6833 (visited on 09/11/2023).

[39] Bitpay Inc. *Accept Bitcoin Payments with the #1 Crypto Payments Processor | BitPay*. 2023. URL: https://bitpay.com/business/ (visited on 09/02/2023).

[40] Bitpay Inc. *Verifone Adds BitPay to Payment Terminals for Purchases In-store, In-App and Online*. 2021. URL: https://bitpay.com/blog/verifone-adds-bitpay-to-payment-terminals-for-purchases-in-store-in-app-and-online/ (visited on 09/11/2023).

[41] S. Eskandari, J. Clark, and A. Hamou-Lhadj. "Buy your coffee with bitcoin: Real-world deployment of a bitcoin point of sale terminal". In: *2016 Intl IEEE Conferences on Ubiquitous Intelligence & Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld)*. IEEE. 2016, pp. 382–389.

[42] A. G. Khan, A. H. Zahid, M. Hussain, and U. Riaz. "Security Of Cryptocurrency Using Hardware Wallet And QR Code". In: *2019 International Conference on Innovative Computing (ICIC)*. 2019, pp. 1–10. DOI: 10.1109/ICIC48496.2019.8966739.

[43] M. Froehlich, J. A. Vega Vermehren, F. Alt, and A. Schmidt. "Implementation and Evaluation of a Point-Of-Sale Payment System Using Bitcoin Lightning". In: *Nordic Human-Computer Interaction Conference*. NordiCHI '22. Aarhus, Denmark: Association for Computing Machinery, 2022. ISBN: 9781450396998. URL: https://doi.org/10.1145/3546155.3546700.

[44] D. Li, W. E. Wong, M. Chau, S. Pan, and L. S. Koh. "A survey ofNFC mobile payment: Challenges and solutions using blockchain and cryptocurrencies". In: *2020 7th International conference on dependable systems and their applications (DSA)*. IEEE. 2020, pp. 69–77.

[45] A. Voskobojnikov, O. Wiese, M. Mehrabi Koushki, V. Roth, and K. Beznosov. "The u in crypto stands for usable: An empirical study of user experience with mobile cryptocurrency wallets". In: *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. 2021, pp. 1–14.

[46] S. Ghosh, J. Goswami, A. Kumar, and A. Majumder. "Issues in NFC as a form of contactless communication: A comprehensive survey". In: *2015 International Conference on Smart Technologies and Management for Computing, Communication, Controls, Energy and Materials (ICSTM)*. IEEE. 2015, pp. 245–252.

[47] N. E. Tabet and M. A. Ayu. "Analysing the security of NFC based payment systems". In: *2016 International Conference on Informatics and Computing (ICIC)*. IEEE. 2016, pp. 169–174.

[48] M. Bellare, J. A. Garay, R. Hauser, A. Herzberg, H. Krawczyk, M. Steiner, G. Tsudik, E. Van Herreweghen, and M. Waidner. "Design, implementation, and deployment of the iKP secure electronic payment system". In: *IEEE Journal on selected areas in communications* 18.4 (2000), pp. 611–627.

[49] S. Kungpisdan, B. Srinivasan, and P. D. Le. "A secure account-based mobile payment protocol". In: *International Conference on Information Technology: Coding and Computing, 2004. Proceedings. ITCC 2004.* Vol. 1. IEEE. 2004, pp. 35–39.

[50] S. J. Grossman and O. D. Hart. "An analysis of the principal-agent problem". In: *Foundations of Insurance Economics: Readings in Economics and Finance*. Springer, 1992, pp. 302–340.

[51] Etherscan. *Ethereum Average Gas Price Chart | Etherscan*. 2023. URL: https://etherscan.io/chart/gasprice (visited on 09/11/2023).

[52] BitInfoCharts. *Bitcoin Avg. Transaction Fee Chart*. 2023. URL: https://bitinfocharts.com/comparison/bitcoin-transactionfees.html (visited on 09/11/2023).

[53] Solana Foundation. *Durable Transaction Nonces*. 2023. URL: https://docs.solana.com/implemented-proposals/durable-tx-nonces (visited on 09/11/2023).

[54] Advanced Card Systems Ltd. *NFC Contactless Payments - ACR122U USB NFC Reader | ACS*. 2023. URL: http://www.acs.com.hk/en/products/3/acr122u-usb-nfc-reader/ (visited on 09/11/2023).

[55] Solana Foundation. *Web3 JavaScript API*. 2023. URL: https://docs.solana.com/developing/clients/javascript-api (visited on 09/11/2023).

[56] Decaf Inc. *@Simgesteen from @GainForestNow paying for coffee with @solanapay at @hackerhouses*. 2023. URL: https://twitter.com/Decaf_so/status/1650520470142238721?s=20 (visited on 09/11/2023).

[57] Decaf Inc. *Decaf coming to Saga @solanamobile*. 2023. URL: https://twitter.com/Decaf_so/status/1649899887608098817?s=20 (visited on 09/11/2023).

[58] The MystenLabs Team. "The Sui Smart Contracts Platform". In: (2022). URL: https://docs.sui.io/paper/sui.pdf (visited on 09/11/2023).

[59] Aptos Foundation. "The Aptos Blockchain: Safe, Scalable, and Upgradeable Web3 Infrastructure". In: (2022). URL: https://aptos.dev/assets/files/Aptos-Whitepaper-47099b4b907b432f81fc0effd34f3b6a.pdf (visited on 09/11/2023).