In Wallet We Trust: Bypassing the Digital Wallets Payment Security for Free Shopping

Raja Hasnain Anwar
University of Massachusetts Amherst
ranwar@umass.edu

Syed Rafiul Hussain
Pennsylvania State University
hussain1@psu.edu

Muhammad Taqi Raza
University of Massachusetts Amherst
taqi@umass.edu

Abstract

Digital wallets are a new form of payment technology that provides a secure and convenient way of making contactless payments through smart devices. In this paper, we study the security of financial transactions made through digital wallets, focusing on the authentication, authorization, and access control security functions. We find that the digital payment ecosystem supports the decentralized authority delegation which is susceptible to a number of attacks. First, an attacker adds the victim's bank card into their (attacker's) wallet by exploiting the authentication method agreement procedure between the wallet and the bank. Second, they exploit the unconditional trust between the wallet and the bank, and bypass the payment authorization. Third, they create a trap door through different payment types and violate the access control policy for the payments. The implications of these attacks are of a serious nature where the attacker can make purchases of arbitrary amounts by using the victim's bank card, despite these cards being locked and reported to the bank as stolen by the victim. We validate these findings in practice over major US banks (notably Chase, AMEX, Bank of America, and others) and three digital wallet apps (ApplePay, GPay, and PayPal). We have disclosed our findings to all the concerned parties. Finally, we propose remedies for fixing the design flaws to avoid these and other similar attacks.

1 Introduction

Digital wallets (e.g., ApplePay, GPay, and PayPal) leverage advanced technological capabilities that revolutionize the traditional payment ecosystem. They allow instant and frictionless payments through customers' smart devices. Digital wallets are designed to be secure by integrating authentication, authorization, and access control security functions [7, 9, 47]. Because of these added security benefits and ease of payment, they will become the primary mode of payment in the near future [48, 69]. A recent research study [55] has found that the total number of digital wallet users will exceed 5.2 billion globally in 2026, up from 3.4 billion in 2022.

Table 1: *Summary of card lock policies for major US banks. Some banks allow (✓) certain types of transactions on locked cards while blocking (✗) others.*

Card Issuer Banks	Physical Card	Wallet (one-time)	Wallet (Recurring)
AMEX	✗	✓	✓
Chase	✗	✓	✓
Discover	✗	✓	✓
US Bank	✗	✗	✓
Citibank	✗	✗	✓
BoAmerica	✗	✗	✓

The integration of new technology into our daily financial transactions has undoubtedly improved convenience, but it has also exposed us to new vulnerabilities. Yet, while the systems supporting digital wallets have been widely embraced by industry and users alike, systematic security analyses for different aspects of digital wallets, e.g., stolen physical cards linked with a digital wallet app are still in their infancy. As digital wallets require sensitive personal and financial information, such as credit card numbers and bank account details, failure to reason about and ensure the security of these systems leads to identity theft and financial fraud [49].

The digital payments ecosystem consists of five notable entities: cardholder, digital wallet, merchant, point-of-sale (POS) terminal, and bank. To facilitate financial transactions, these entities establish trust in other entities by delegating specific tasks. For instance, the bank entrusts the digital wallet with user authentication and cardholder verification.

The security of the payment protocols running between the device and the POS has been investigated in the past [12, 44, 68, 70]. However, the protocols involving different entities in the digital payments ecosystem have received little attention mainly due to the system complexity, lack of formal documentation, and closed-source wallet implementations. This motivates us to perform a research study regarding how

Table 2: *Summary of key findings of our study on the decentralized authority for the digital payments ecosystem.*

Capability	Vulnerabilities	Attacks	Root Cause	Proposed Solution
Authentication bypass	Wallet's KBA-based authentication over MFA	Add stolen card into wallet	Delegation of the authentication method choice to wallet	Push-based MFA authentication method
Cardholder verification bypass	In-device wallet authentication used instead of PIN & signature	Authorize transaction using own device and wallet	Unconditional trust in wallet-based verification	Continuous authentication for token updates
Access control violations	No checks for recurring payments	One-time payments of arbitrary amounts	Trap door to bypass access control restrictions	Check payment metadata

digital payments are treated when the card is locked by the cardholder, and/or reported as stolen to the bank (see Table 1). In particular, we conduct a study to answer the following research questions:

RQ1 on authentication. What is the effectiveness of the security measures enforced by the bank and digital wallet for adding a card to the digital wallet?

RQ2 on authorization. How can an attacker make payments using a stolen card through a digital wallet?

RQ3 on authorization. Do victim actions, i.e., **(a)** locking the card, and **(b)** reporting the stolen card and requesting a replacement, suffice to prevent malicious payments on stolen cards?

RQ4 on access control. How can an attacker bypass the access control restrictions on stolen cards?

To answer these questions, we conduct an empirical assessment where we consider practical use case scenarios. Our strategy is to observe the digital wallets payment for: (i) different events (card lock, card loss reported to bank); (ii) arbitrary amounts (payments of large vs. small amounts); (iii) different ways (online, in-store, merchant app); (iv) distinct methods (physical card vs. digital wallets); (v) and payment types (one-time vs. recurring).

Our study identifies the vulnerabilities in the decentralized authority for payments through digital wallets. It reveals that the attacker can bypass the core security functions by exploiting the trust relationship of different elements in the digital payments ecosystem. Table 2 summarizes our findings in terms of revealed issues in the payments ecosystem and provides the recommended fixes.

To make payments using digital wallets, the user first needs to add the card to the wallet. The `CARD ADD` procedure triggers user authentication with the bank. Once authenticated, the user can perform both online and in-store transactions by using their digital wallet. As part of the authentication process, both the wallet and the bank need to agree on an authentication method (i.e., knowledge-based authentication (KBA) or multi-factor authentication (MFA)). The wallet sends the choice of its supported methods in the order of preference to the bank. Instead of choosing the relatively secure authentication method, the bank selects the one marked with the higher priority by the wallet; hence it ends up using the least secure authentication method (e.g., easy-to-guess ZIP code-based KBA [4, 17]) for `CARD ADD` procedure. The attacker exploits this vulnerability and adds the victim's card into the wallet (addressing *RQ1*).

It is practical to assume that when the victim finds their card missing or lost, they will (a) lock the card, and/or (b) report to the bank in order to receive a new card. Meanwhile, the bank revokes this physical card's payment authorization to avoid fraudulent activities. However, our study finds that payments (both online and in-store) initiated through digital wallets are not blocked by certain banks (refer to Table 1). This is because once the cardholder is authenticated, the bank establishes an *unconditional trust* with the wallet. It substitutes the wallet's biometric security (e.g., facial recognition, and fingerprint verification) for cardholder verification, which is part of the payment authorization process. The attacker exploits this vulnerability and bypasses the payment authorization at the bank. They use their own wallet to perform in-device biometric authentication, but make purchases through the victim's stored card (which is locked/reported) in the wallet (answering *RQ2*).

Apart from making in-store and online payments, a user can also set up recurring payments for subscription-based services (e.g., Apple Music). Recurring payments are different from one-time payments, such that a fixed amount is periodically paid to the merchant. For these payments, the bank issues a `contractID` to the merchant which is used to post the recurring transactions. Our study finds that the access control restrictions, that are applied to one-time payments, do not exist for recurring transactions. This choice is by design where the card issuing banks explicitly state that recurring payments will be authorized despite the card being locked. Table 1 shows that one-time transaction restrictions on locked cards do not apply to all banks. We argue that this design choice opens a trap door for violating the access control restrictions for one-time payments. The attacker registers for a subscription by using the victim's card in the wallet, but makes a one-time transaction. They can use this technique to receive services that are not even defined as subscription-based services, like booking a hotel room, and renting a car. Even worse, they can successfully make

purchases of an arbitrary amount, and at any time of the day (which violates the subscription contract registered with the `contractID` at the bank). This answers *RQ3*.

Validations. We validate our findings by launching practical attacks. We consider a passive attacker who is *not* capable of modifying or tampering with the communication between different elements of the digital payments ecosystem. We consider major US banks, and prominent digital wallet apps designed for both iOS and Android. We demonstrate the attacker's ability to make purchases by using the victim's bank card which is locked or reported as stolen to the card issuing authority. The attacker can buy different goods and services by using their digital wallet app for (a) in-store, (b) online, and (c) merchant app transactions. Table 2 shows the summary of our findings where we have tried a number of *locked* bank cards issued by major US banks, and made financial transactions through different modes of payments.

Ethics and Disclosure. We are mindful that some of our tests and attack evaluations might be damaging to financial institutions and merchants. We thus conduct the evaluations in a responsible manner through two measures. Firstly, we use our own credit cards and wallet accounts for both the attacker and the victim. Secondly, when evaluating the attacks in practice, we do not claim or report any of the exploited transactions to the bank; hence bear all the financial costs of the purchases and do not defraud the bank, wallet, or merchant.

We initially disclosed our findings to all discussed US banks and digital wallet providers in April 2023. Chase, Citi, and Google have responded to us. They have acknowledged that the discussed vulnerabilities can be exploited under certain conditions (as detailed in this paper). We followed up with the banks and digital wallet providers in February 2024 regarding the mitigation status of the reported vulnerabilities. We received responses from Google, Citi, Chase, and Discover. At the time of writing this paper, Google is working with the banks from its end to address the reported issues on Google Pay. The banks, however, reported to us that the disclosed attacks are not possible anymore. Chase confirmed that additional fraud detection and transaction limitation measures have been put in place to address the reported vulnerabilities; Citi and Discover, however, did not disclose the specific mitigation measures to us. We did not yet receive responses from AMEX, BoA, US Bank, Apple, and PayPal.

Organization. In Section 2, we provide technical background on security procedures in digital payments, covering user authentication, cardholder verification, and access control. Section 3 delves into various digital wallet-specific features provided by banks that could be exploited by malicious users. In Section 4, we discuss our methodology and attacks on security procedures surrounding digital payments through wallets. In Section 5, we propose countermeasures to enhance

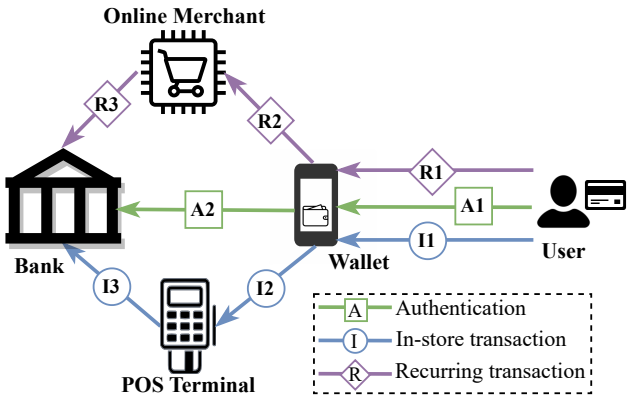


Figure 1: Overview of Digital Payment Communication.

user authentication and transaction authorization procedures throughout the payment lifecycle. Section 6 explores previous works on exploiting digital payments and EMV standards, and finally, we conclude our findings in Section 7.

2 Background on Security Provisioning in Digital Payments Ecosystem

The digital payments ecosystem involves various stakeholders. As the fundamental stakeholder, the bank issues payment cards to their customers – called cardholders. Cardholders add these cards to digital wallet apps (e.g. ApplePay, GPay, and PayPal), allowing them to make contactless payments directly from their smartphones. Figure 1 illustrates primary entities within the ecosystem and their interactions in various transaction scenarios.

Digital wallets support two types of transactions: in-store and online – these are also called one-time payments. For in-store transactions, merchants use point-of-sale (POS) terminals connected to the bank's payment network. Conversely, online transactions occur on the merchants' e-commerce websites. Apart from one-time transactions, wallet users can also establish online recurring transactions with merchants for periodic payments, e.g., monthly subscriptions.

EMV® protocol. Europay, Mastercard, and Visa (EMVCo.) is a global organization that develops and maintains the standard protocol for digital card payments. Its primary objective is to provide a secure and dependable protocol to accommodate the increasing demand for digital payments worldwide. The EMV® protocol for contactless payments [25] lays out multiple security measures designed for each stakeholder entity to ensure a secure communication channel.

The EMV® protocol places the digital wallet at the core of the modern payments system. The wallet-centric system hides the user and card information from merchants to pre-

vent identity theft and fraud. For instance, when a card is added to a digital wallet, the bank issues a payment token to replace the card's primary account number (PAN)[1]. For all transactions conducted with the wallet, the payment token is used instead of the PAN to shield the original card from malicious merchants. Furthermore, the wallet app secures itself behind the smartphone's biometric security features, e.g., facial recognition and fingerprint verification. Therefore, secure communication between the wallet, merchant, and bank is contingent upon the wallet's security.

We refer to the EMV® specifications [24–28, 32] to outline the security procedures involving the digital wallet: user authentication, cardholder verification for transaction authorization, and access control.

2.1 User Authentication in Wallet

When a cardholder adds a card to the wallet, the card issuer bank relies on the wallet for authentication. Since the cardholder is directly interacting with the wallet (A1 in Figure 1), the bank selects an authentication method that aligns with the wallet's capabilities. Generally, two types of authentication methods are used: knowledge-based authentication (KBA), and multi-factor authentication (MFA).

Knowledge-based authentication (KBA) works on the "something you know" principle. It requires the user's personal information like billing ZIP (postal) code, billing address, date of birth, and last four digits of social security number (SSN)[2] for verification [16, 19, 53]. The most prevalent KBA method uses the billing address and ZIP code [20].

Two-factor authentication (2FA) using a one-time password (OTP) is a common MFA method. It is based on the "something you have" principle, where the user possesses a device and an account for receiving the OTP. The specific implementation of the 2FA method varies depending on the wallet service provider: some wallets offer OTPs via SMS, while others use email.

After user authentication, the bank uses a token service provider (TSP) to create and manage tokens linked to the primary account number (PAN). The bank shares the PAN-associated tokens with the wallet (A2 in Figure 1). The wallet does not store the PAN; instead, it uses the token for all transactions, enhancing security for card and user data.

2.2 Cardholder Verification

Cardholder verification (CV) is a pivotal part of transaction authorization. It ensures that only legitimate cardholders can make transactions on Point-of-Sale (POS) terminals. POS terminals support a number of cardholder verification methods (CVMs) applicable to all types of cards and wallets. These CVMs include signature, offline plaintext PIN, offline enciphered PIN, online PIN, offline plaintext PIN and signature, offline enciphered PIN and signature, and consumer device CVM (CDCVM) [24].

In consumer device CVM (CDCVM), the cardholder verifies the wallet-based transaction on their smartphone using either a passcode, pattern, or biometric ID. It aligns with the "something you have" principle, allowing any user with an authenticated wallet app on their smartphone to complete payments without needing additional identification like a PIN or signature. Nevertheless, the user is still required to confirm their device ownership (I1 in Figure 1). For this, the user unlocks their smartphone and the wallet app, and the POS terminal reads the token from the unlocked wallet app (I2 in Figure 1). POS forwards the token and transaction details to the bank (I3 in Figure 1) to complete the transaction.

2.3 Access Control

When a transaction reaches the bank for authorization, it undergoes a verification process to confirm the cardholder's eligibility. The bank uses an access control service to assess transactions based on criteria like transaction type, amount, time, and card status. Additionally, the bank's access control service conducts fraud detection by comparing the transaction against similar historical transactions. This ensures that the transaction aligns with the cardholder's typical spending behavior and is not unusual.

The bank adopts distinct access control policies for different types of transactions. For one-time – online and in-store – transactions, the access control ensures that the user account has enough balance to cover the payment. However, for recurring transactions, an additional condition is imposed: the card must be associated with a credit account (R1 in Figure 1) – most banks do not allow recurring transactions on debit cards.

Given that the bank places varying levels of trust in the entities within the ecosystem, it only allows transactions from trusted sources during times of heightened fraud risk. For instance, when the cardholder locks their card due to loss or theft, the bank continues to allow transactions from digital wallets (R2–R3 and I2–I3 in Figure 1). Simultaneously, it blocks all in-store and online transactions made using the physical card. It is worth noting that specific access control policies may vary between banks, as highlighted in Table 1. While some banks do not allow one-time transactions after card lock, all banks universally permit recurring transactions.

3 Features or Vulnerabilities?

Digital wallets are considered to be more secure than traditional transaction methods [7, 9, 47]. Wallet's device-based biometric security mechanisms facilitate the bank in user authentication and cardholder verification [25, 28]. Because

[1] PAN is the 15- or 16-digit number on the front of the payment card

[2] In the US, SSN is a nine-digit ID used for earnings and taxation reporting.

Table 3: *Summary of intended features that can be exploited by the adversary.*

Features	Rationale	Vulnerabilities	Exploit
Multi -device & -user card access through wallets	Ubiquitous card access through wallet	Attacker adds stolen card to their wallet posing to be legitimate user	Weak KBA authentication to add stolen card to wallet
In-device method used for cardholder verification	Wallet is always possessed by legitimate user	Attacker is verified through their own wallet (device)	Stolen card used for transactions without PIN and signature
Uninterrupted recurring payments	Always allow recurring transactions to avoid late payment fees	Merchant labels one-time transactions as recurring	Mislabeled transactions bypass card lock restrictions

of such a strong authentication guarantee, the bank provides payment features to wallet users that are not accessible for physical cards. The backbone of these features is the bank's trust in wallet security mechanisms. We demonstrate that these features can be exploited into several security vulnerabilities. Our findings are summarized in Table 3.

1. *Multi-device and multi-user card access through wallets give rise to weaker authentication.* The bank allows a cardholder to link their card to multiple devices (e.g., smartphone and smartwatch). The cardholder can also enable digital card access to their family members' devices; meaning that a card can be added to multiple wallets from the same provider, like ApplePay, but with different account IDs, such as Apple IDs of different users. For instance, an AMEX card can be linked with up to 10 Apple Pay accounts simultaneously [5].

To implement this feature of ubiquitous access and easy sharing of cards across devices, the bank and wallet service provider assume that the cardholder and device/wallet owner is the same person. Based on this assumption, the bank and the wallet often opt for weaker authentication schemes. The attacker can, however, turn this feature into a vulnerability and exploit it to add a stolen card to their digital wallet.

2. *The wallet verification in lieu of the cardholder verification is flawed.* Instead of performing a separate cardholder verification procedure for wallet users, the bank relies on in-device biometric verification methods to identify the cardholder authorizing transactions. Although it makes the payment authorization procedure convenient and quicker, it leads to a security vulnerability where the attacker can falsely claim to be the cardholder.

3. *The access control restrictions do not apply to recurring transactions.* The banks allow payments for subscription-based services even on lost/stolen cards in order to shield the cardholder from late payment fees/penalties. The malicious user can exploit this feature (vulnerability!) and make the one-time transaction as if it is a recurring payment, and bypass the transaction authorization restrictions at the bank.

Are banks aware of these flaws? We have shared our findings with major US banks and wallet providers. Two of them have responded and also acknowledged that these features can inadvertently give rise to security vulnerabilities under certain conditions. The following section describes the attack scenarios possible through unfettered wallet access.

Threat Model. The presumed attacker is a digital wallet user, whereas the victim is a cardholder possessing a credit card issued by a major US bank. We assume the adversary to be a passive attacker who cannot disrupt the communication between the bank and wallet. That is, we consider the channel between the wallet and the bank provides authenticity and confidentiality.

We also assume that the attacker can obtain a stolen or lost card of a victim, and the victim has not yet reported the incident to the corresponding card issuer bank. This is a realistic assumption as there is a gap between the time the card is stolen and the time the victim realizes it and reports it to the bank. The attacker can exploit this time window to launch the attacks. Similar assumptions have also been made in related work [13].

A stealthy attacker can go undetected. To avoid being detected by the bank, the attacker does not attempt to make any in-store or online purchases with the stolen physical card. The attacker can utilize various other security measures to obscure their identity and evade detection by both banks and wallet providers. Fake identities [33, 36, 64] and burner phones [15,21,29,38] are standard tools for impersonation and fraud attacks [57, 61]. Prior research has shown that attackers can build fake identities using virtual phone numbers and fake emails, and conduct fraudulent banking activities [1, 34, 67]. Along the lines of these works, we assume that an attacker employs these mechanisms to remain stealthy and prevent the banks and wallets from tracing them.

4 Insecurities in Decentralized Authority for the Digital Payment Ecosystem

Methodology. Our vulnerability identification methodology follows a three-step process. Firstly, we identify token lifecycle management events [27, 28] and their corresponding procedures governing digital payments. These procedures detail how the binding between PAN and token operates during the lifecycle management events. We particularly focus on critical events such as token activation (`CARD ADD`), token suspen-

sion (card loss), token attributes update (`LifecycleUpdate`), token detachment (device loss), and token provisioning for different types of payments.

Secondly, we analyze the payment procedures identified in the first step. We consult EMV® specifications [24–28, 32] and digital wallet documentation [6, 16, 19, 51] to study the participating entities and the data exchanged between them. The objective is to understand the intended functionality of the procedures and then potentially identify flaws and weaknesses within them. This includes identifying: (i) trust assumptions between the entities, (ii) differences between procedures using PAN and those using tokens, (iii) inadequate/inconsistent security policies, and (iv) flaws in security policy enforcement. We analyze high-level security policies for critical payment procedures: authentication, authorization, and access control.

Lastly, we assess the feasibility of launching practical attacks on the weaknesses found in the second step. In this regard, we validate the intended functionality of procedures against how banks and wallets implement them in practice. We examine the policies of major US banks regarding payment processing, particularly in situations of elevated risk. These scenarios include when the cardholder: (i) locks the card, and (ii) reports the card loss and orders a replacement. This step underscores any weaknesses in the EMV specifications that persist in bank policies, potentially enabling attackers to exploit them to launch attacks using a stolen card.

High-level overview of our findings. Our experiments reveal that banks indeed delegate the control of certain procedures to digital wallets and opt to use weaker security mechanisms to provide additional features to wallet users. Therefore, the attacker can launch several attacks by exploiting vulnerabilities rooted in the authentication, authorization, and access control procedures of the digital payment ecosystem. Refer to Table 2 that summarizes our findings.

The attacker breaks authentication by forcing the bank to fall back to a weaker authentication procedure for linking the card to the digital wallet. In consequence, they can successfully add a stolen card to their own digital wallet.

Upon authentication, the bank provides a card token to the wallet for payment authorization. We find that this token neither expires nor gets updated despite the card being locked, or reported stolen (where the bank issues a new card). This enables the attacker to make purchases using the victim's card added to the digital wallet, despite the bank canceling the stolen card and issuing a new one.

The bank implements a transaction-type access control policy in which recurring payments are always allowed while one-time transactions can be restricted after the card is locked/lost. We find that such access-control checks are regulated by merely inspecting the transaction label (recurring vs non-recurring) rather than validating the complete transaction data (e.g., purchase type, time, amount, etc.). The attacker exploits this flaw by labeling their one-time transaction to be recurring, and bypasses the access-control policy. In consequence, they can make purchases of arbitrary amounts, despite the bank's policy to block these transactions.

Are the vulnerabilities time-dependent on card lock? The impact of the vulnerabilities is not bound to the timing of the attack relative to the card lock. The bank enforces the card lock restrictions as soon as the card is locked, and these restrictions remain consistent for the duration of the card lock. Following our threat model, the attacker needs to add the card *before* the victim locks it. After adding the card, the attacker does not need to time the attacks precisely at the time of card lock. This is because the vulnerabilities do not arise during time-of-check to time-of-use (TOCTOU), but are rooted within the policies adopted by the bank. We conduct repeated experiments at different intervals following the card lock to verify this.

Who incurs the financial cost? We discover that these attacks are of a serious nature and remain unnoticed by the bank, wallet provider, and merchant until the victim disputes the charges. When the victim disputes the fraudulent transaction, the merchant is more liable to pay for the loss than the bank [35, 39]. However, a non-vigilant victim may not check the credit card statements regularly, and hence will not even notice being under attack.

4.1 Trust in Digital Wallets Weakens User Authentication

The digital wallet is a functional entity that connects the cardholder with the bank. It facilitates the bank in user authentication procedures and token lifecycle management. This fosters the bank's trust in the digital wallet where the bank (i) allows the wallet to choose the authentication method, and (ii) automatically updates replacement cards in the wallet without requiring user re-authentication.

4.1.1 Wallet-Driven Authentication Method Selection

The bank delegates the choice of user authentication method to the wallet. The wallet performs authentication by presenting a list of its supported methods (e.g., billing address, OTP, and call) to the user. This means that it is the end-user who makes the final decision regarding the authentication method to be used for adding the card to the wallet. Such delegation of authentication is flawed in that an attacker can dictate the bank to accept a weak authentication procedure which gives birth to a number of security vulnerabilities.

> **Issue 1:** Delegation of authentication from the bank to digital wallets allows an attacker to add a stolen card to their wallet and make transactions.

Attack details. Figure 2 describes an attack scenario where the attacker compromises the authentication process and adds

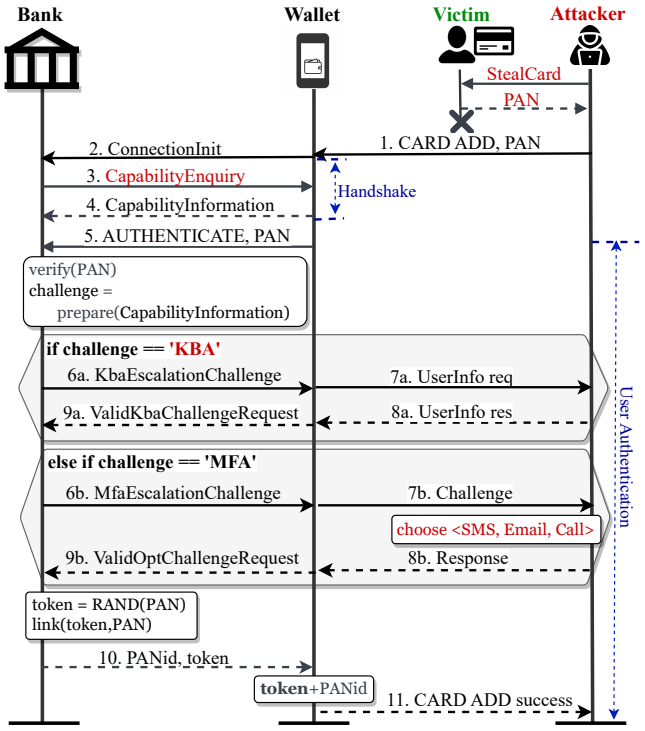


Figure 2: The authentication procedure in the digital wallet; the attacker exploits weaknesses in KBA- and MFA-based authentication to add the victim's card to their wallet. The authentication process comprises two phases: a) the handshake between the wallet and the bank, and b) the user authentication in the wallet delegated by the bank. Weaknesses in the process, highlighted in red, allow the attacker to add the card.

the victim's card to their (attacker's) wallet. Following our threat model, we consider that the victim has lost their card and has not yet locked it. Meanwhile, the attacker is able to get their hands on the victim's card. The attacker starts the `CARD ADD` process by inputting the card's primary account number (PAN) in the wallet (message ①). It triggers the authentication procedure in which the bank and the wallet execute a handshake (② - ④), and first agree on an authentication method to be used.

The bank sends a `CompatibilityEnquiry` request (③) to get the wallet's supported authentication methods. The wallet responds with a list of supported methods in the order of preference in a `CompatibilityInformation` message (④). Upon receiving the supported authentication methods, the bank chooses the one with the highest preference. Refer to Section 10.5 in EMVCo. Book 3: Application Specification [24] for more details on this procedure.

The wallet sends an `AUTHENTICATE` request (⑤) containing the victim's PAN to the bank. Once the bank receives the authentication request from the wallet, it verifies the PAN and generates the authentication response. For simplicity, in Figure 2, we consider two widely used authentication methods: *knowledge-based authentication (KBA)*, and *multi-factor authentication (MFA)*, that are widely used by major banks in the US. The bank sends `KbaEscalationChallenge` (⑥a) to the wallet when KBA is used; otherwise, it responds with `MfaEscalationChallenge` (⑥b) if MFA is used as an authentication method.

In KBA, the wallet receives the `KbaEscalationChallenge` from the bank (⑥a). It prompts the user (⑦a) to verify their identity by providing one or more information items: the billing ZIP code, billing street address, date of birth, and/or the last four digits of SSN [53]. The wallet sends the user information to the bank via `ValidateKbaChallengeRequest` (⑧a – ⑨a). The bank completes the authentication procedure once the user is verified through address verification service (AVS) [16, 19, 51].

In the case of MFA, the wallet receives `MfaEscalationChallenge` from the bank (⑥b) and prompts the user to choose their preferred authentication method (⑦b). The user chooses either to receive a one-time password (OTP) via SMS or email, or call the bank to verify their identity. If the user chooses OTP, the wallet sends `ValidateOtpChallengeRequest` message [3] and completes the user authentication (⑨b) by validating the OTP(s). In call-based authentication, the user provides their date of birth or last four digits of SSN to the bank for identification – note that this is similar to KBA.

Once the user is authenticated, the bank generates a token (a proxy identity for PAN) using the token service provider (TSP)[4]; and shares the token along with associated last four digits of the PAN, called `PANid`, with the wallet (⑩). The authentication procedure completes when the wallet sends a `CARD ADD` *`success`* notification (⑪) to the user.

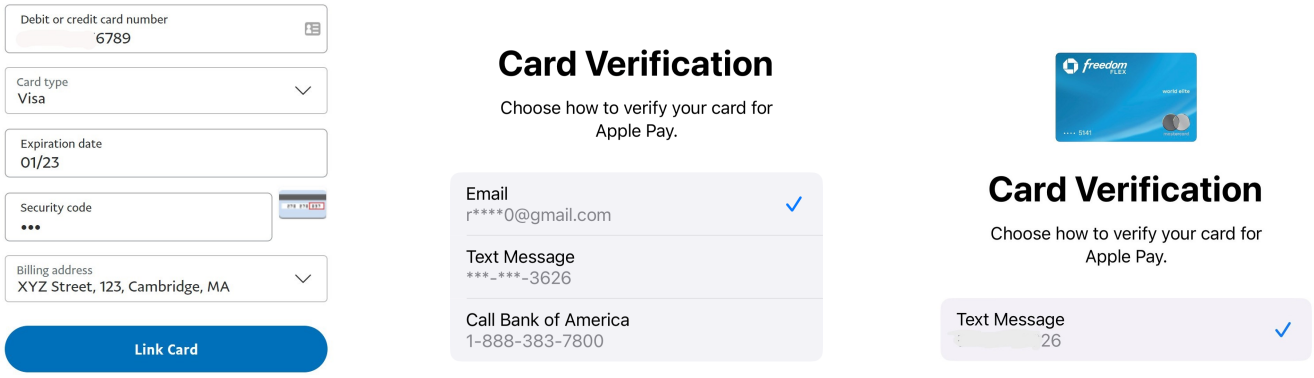
Key takeaways. The end-user, but not the bank, decides the authentication method to be used. For example, an attacker can make the bank *fall back* to KBA when MFA is mandated. They do so by using the "call-based" authentication option. The attacker dials the bank's automated helpline for adding the card to the wallet. The helpline prompts the attacker to provide the KBA-related information: date of birth and the last four digits of SSN associated with the victim's card.

The wallet uses KBA-based authentication methods because they are convenient and easy to use for customers. By prioritizing ease of use, the bank essentially fails to enforce a higher standard of security. This is because KBA methods are proven to be much weaker than MFA-based methods [4, 17].

The attacker can acquire the information needed for

[3] `POST /users/{user_id}/sendOtp` endpoint remains open for the wallet to request additional OTPs.

[4] TSP links PAN to the token that wallet uses as card ID for transactions.



(a) BoA in PayPal (b) BoA in ApplePay (c) Chase in ApplePay

Figure 3: Authentication methods used in different wallets. The BoA credit card uses (a) KBA for the Paypal wallet, and (b) multiple MFA methods (i.e., email, SMS, or the call) for ApplePay as the authentication methods. In contrast, for Chase bank credit card, (c) ApplePay offers only SMS-based MFA. These experiments highlight the inconsistencies in authenticating the user not only within the wallet (i.e., Apple Pay), but also across the wallet apps (i.e., Apple Pay and Paypal).

KBA authentication from the publicly available online databases [3, 46, 50, 60]. For instance, ZIP code and date of birth are practically considered public information, especially in the age of social media [18, 62]. Therefore, the attacker uses non-trivial methods to acquire the victim's personal information and adds the stolen card to the wallet which uses KBA authentication – the weakest authentication method.

Empirical assessment. We evaluate the efficacy of adding a victim's stolen card to an attacker's wallet, irrespective of the wallet the victim is using. Our experiment setup includes two credit cards issued for the victim by the Bank of America (Visa card) and the Chase bank (Master card), respectively; the attacker steals these cards. We consider both smartphone devices (an Android device for PayPal and GPay wallets, and an iPhone for ApplePay) to be under the control of the attacker and are not rooted. For a realistic attack setting, we ensure that both the credit cards and the smartphones have no collective use history. Furthermore, both the attacker and the victim do not share any common information (e.g., billing address).

We find that PayPal and GPay use only the billing address for KBA (Figure 3 (a)), whereas the ApplePay uses two different sets of methods for MFA (Figure 3 (b) & (c)) for each card. For the Bank of America Visa credit card, ApplePay provides three MFA options: SMS OTP, email OTP, and verification over automated call (Figure 3 (b)). However, it uses only SMS OTP as MFA for the Chase Master credit card (Figure 3 (c)).

The attacker uses popular online databases, like Yellow Pages[5] and SearchBug[6], to acquire the victim's last four digits of SSN, date of birth, and ZIP code – commonly used for KBA. Similarly, there are several past studies [42, 43] that demonstrate the weaknesses in the SMS-based OTP. We rely on these works to highlight the feasibility of adding the card through the MFA-based authentication procedure.

Further, we find that the delegation of choice for authentication method leads to an inconsistent authentication policy. One bank card can be authenticated in multiple different ways in different wallets (Figure 3 (a) vs. (b)). Yet, the authentication method does not vary with the wallet provider that the attacker and the victim are using. For example, it is not more difficult for the attacker to use PayPal if the victim has already added the card to their PayPal wallet.

Root causes. The above experiment demonstrates that some digital wallets, like PayPal, prefer KBA over MFA. They consistently prioritize the user-friendly KBA-based authentication methods, regardless of the card issuer bank. Even if some wallet uses call-based authentication (which it considers MFA), it falls back to KBA.

The bank does not choose the authentication method of higher security, but rather it honors the wallet's preferred choice. With weak authentication, an attacker can steal a card and add it to their wallet for transactions.

Lessons learned. Although the delegation of authority for authentication is efficient and scalable, it compromises security. This becomes evident when crucial decisions are also delegated to third parties in the digital payment ecosystem. We identify that a foolproof and uniform authentication policy enforcement by the bank is missing for all wallets.

4.1.2 Card Replacement in Wallet Without User Authentication

The bank retains the trust chain with the wallet beyond the user authentication. When a change occurs in the status of the linked card (e.g., `DeletionOfPaymentCredential` and

[5] www.yellowpages.com
[6] www.searchbug.com

Lost/Stolen ConsumerDevice), the bank sends the card-related updates directly to the wallet [28]. These updates are conveyed through a LifecycleUpdate request to the linked wallet. Notably, this process does not notify the user of a LifecycleUpdate nor requires the user re-authentication for accepting the update. Thus, the wallet that is under the control of the attacker also receives such updates.

> **Issue 2:** The card replacement event does not revoke the token; allowing the attacker to keep using the previously added card in their wallet without requiring authentication.

Attack details. We consider the lost card scenario to present the token updates without user re-authentication. The complete workflow is illustrated in Figure 4. When the user reports the card loss (①), the bank blocks the lost card and issues a new card (with a new PAN) to the user. However, it does not update the associated token; instead, it links the old token with the new PAN in the token service provider (TSP) [66].

The bank issues a LifecycleUpdate and sends the new PANid to all wallets liked with the lost card (messages ②a and ②b in Figure 4). These wallets replace the stored PANid with new PANid and link it with the stored token. At this point, the same old payment token is linked with the new PANid (and PAN) in wallets as well as TSP.

The LifecycleUpdate automatically adds the new card to the wallet (③a and ③b), and does not verify the user – whether they should receive the card replacement update or not. Because the user has not initiated the CARD ADD procedure, the bank does not trigger EscalationChallenge procedure for adding the replacement card [32].

Key takeaways. One-time user authentication bypasses the authentication for adding a replacement card to the wallet after a card loss event. While an automatic card replacement is convenient for wallet users who do not have to manually add the card, it weakens credit card security. It means once the attacker has added the stolen victim's card to their wallet (refer to Section 4.1.1), they can continue using it despite the victim receiving the replacement card from the bank.

Empirical assessment. We perform several experiments to validate the flawed PAN replacement policy for two major US banks: American Express (AMEX) credit card, and Chase Visa credit and debit cards. In our experiments, all three cards are first added into both the victim's and attacker's wallets (ApplePay, PayPal, and GPay). Once the attacker has successfully added the card to their wallet, the victim reports the card loss event by contacting the bank and requesting the issuance of a new card. The bank immediately blocks the reported card and mails the new card to the victim.

We observe two different outcomes at the attacker wallets.

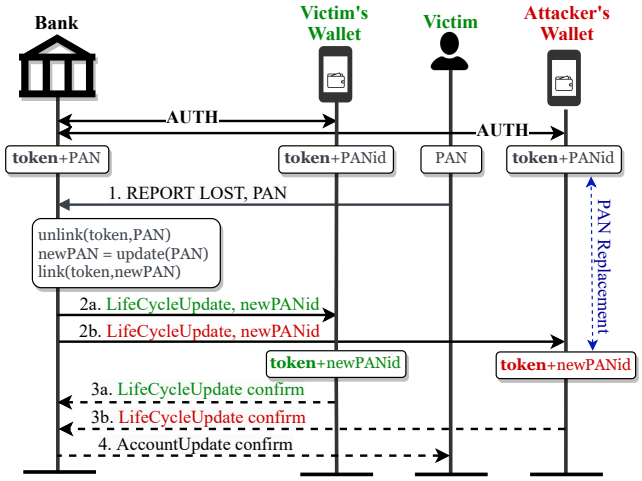


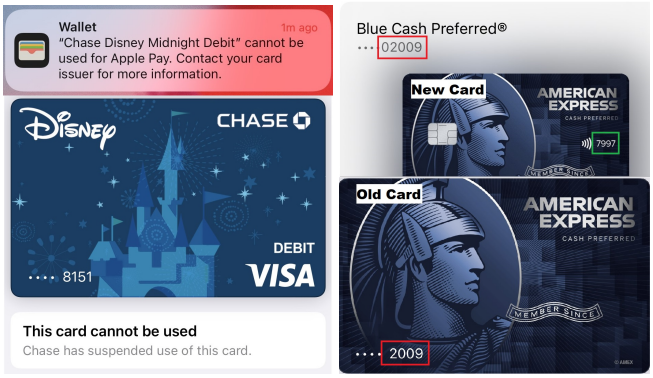
Figure 4: PANReplacement procedure for replacing a card upon user report of card loss. The bank issues a new PAN and updates the new PANid in all the associated wallets without user authentication. Green messages indicate communication with a trusted (victim's) wallet, while red messages indicate a malicious wallet used by the attacker.

First, in the case of credit cards (Chase Visa and AMEX), the attacker's wallets receive a silent update from the bank in which they link the new PANid with the stored token. Second, for the Chase Visa debit card, the wallet sends a notification (as shown in Figure 5 (a)) stating that the debit card cannot be used with ApplePay anymore.

We continue our experiments to validate whether the attacker can use the credit cards in their wallet even *after* the victim reports these credit cards to the bank as stolen. We find that the attacker can indeed make purchases without any disruption (i.e., even before the victim receives the new credit card in the mail). An unusual observation related to this experiment is that albeit the wallet displays the new PANid (*7997*, as shown in Figure 5 (b)), it continues to use the old PANid (*2009*, as shown in Figure 5 (b)). We confirm this where the old PANid has appeared on the merchant receipts for payments made through the wallets.

Root causes. The bank has established an unconditional trust with the wallet. Due to this trust, the bank does not implement a token revocation mechanism to limit the wallet's access to the card in critical scenarios. Hence, the token remains constant when it binds with the PANid. During the card replacement event, the bank replaces the PANid in the wallet without updating the token.

The bank fails to verify whether the wallets receiving the card updates belong to the card owner or not. This impacts the victim whose card is accessible to the attacker despite a new replacement card being issued by the bank.



(a) Chase Debit Card (b) AMEX Credit Card

Figure 5: Card replacement after reported as lost. The bank blocks the debit card (a), rendering it unusable in the wallet. However, for the credit card, (b) the bank issues a replacement card and updates `PANid` in the wallet, but keeps the old token active for payments.

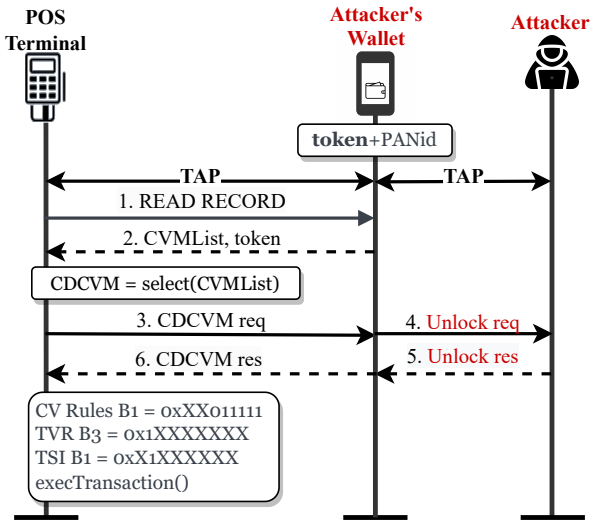


Figure 6: In-store cardholder verification (CDCVM). Messages highlighted in red show weaknesses in the procedure.

Lessons learned. The bank's unconditional trust in the wallet extends beyond the user authentication. During critical events, such as card loss, the bank neither revokes the card's token from the wallet nor requires user verification to continue using the wallet. Therefore, an attacker's wallet receives the same updates as a legitimate user's wallet.

4.2 Unrestricted Transaction Authorization on Wallets

Digital wallets are inherently different and more secure than physical cards [7, 9, 47]; that is why the bank uses two different methods of transaction authorization. For transactions on a physical card, it uses PIN and signature verification. In the case of a wallet, it uses in-device security methods (e.g., facial recognition, and fingerprint verification) to verify the user – known as device cardholder verification method (CDCVM) [24]. The wallet app prompts the user to unlock the device to complete the transaction authorization through an in-device security mechanism. It means that the bank relies on in-device authentication (i.e., CDCVM) for payment authorization instead of performing the traditional cardholder verification. An attacker can bypass the cardholder verification (i.e., through card PIN or signature): they add a stolen card to their wallet (refer to Section 4.1.1), and make the transaction through in-device authentication.

Issue 3: In-device verification method fails to verify both the wallet owner and the cardholder, allowing an attacker to bypass the cardholder verification process.

Attack details. The CDCVM procedure substituting the card owner's verification works as follows. The attacker uses the digital wallet to make an in-store transaction on the POS terminal. As shown in Figure 6, the attacker first taps the device containing their wallet app over the terminal. It requests token data by issuing the `READ RECORD` command to the wallet (①). The wallet responds with the token and CDCVM capability in a list of cardholder verification methods, i.e., CVMList (②).

The terminal requests the wallet to complete CDCVM (③). Wallet prompts the user to unlock the device (④) through fingerprint, face recognition, or device PIN. Once the user unlocks the device, it sends a CDCVM confirmation to the POS terminal (messages ⑤ and ⑥ in Figure 6). With this, the device-based cardholder verification (CDCVM) is complete. Finally, the terminal sends the CVMRules and TSI to the bank for authorization and completes the transaction.

Key takeaways. Since the bank does not require the POS terminal to perform the cardholder verification, the terminal sets the `'No CVM required'` (`0xXX011111`) flag in the CVMRules. It also sets the `'Cardholder verification was not successful'` (B3 = `0x1XXXXXXX`) flag in the terminal verification results (TVR) because the transaction was not verified by the POS. Meanwhile, the wallet transfers the token to the terminal, indicating that the bank has authorized the wallet to make purchases. The terminal proceeds the payment and updates the flag *B3* to `'Cardholder verification was performed'` (B1 = `0xX1XXXXXX`) in transaction status information (TSI). This flag indicates that the cardholder verification was performed in the wallet, but in

Table 4: *Summary of payment transactions of different amounts by using major US banks' locked credit cards.*

Banks	Amount ($)	Physical	Wallet
AMEX	≤ 1	✗	✓
	≥ 500	✗	✓
Chase (VISA)	≤ 1	✗	✓
	≥ 500	✗	✓
Discover	≤ 1	✗	✓
	≥ 500	✗	✓
US Bank (VISA)	≤ 1	✗	✗
	≥ 500	✗	✗
Citi (Mastercard)	≤ 1	✗	✗
	≥ 500	✗	✗
BoAmerica (VISA)	≤ 1	✗	✗
	≥ 500	✗	✗

reality, the device owner's verification was performed (refer to Annex C in EMVCo. Book3: Application Specification [24]).

Empirical assessment. We perform several experiments and launch attacks on in-store purchases made through digital wallets. We consider that the victim's AMEX and Chase Visa credit cards are stolen by the attacker. The attacker adds these cards to their wallet and the victim **locks** both cards. The bank promises to stop the payments on locked cards. The victim receives the following message when they lock the card from the Chase bank app:

"Instantly block new purchases, cash advances, and balance transfers made with your card or card number."

Thereafter, the attacker visits the merchant stores (we test at Walmart, and Target stores), and uses the victim's card stored in their wallet (we validate on ApplePay and GPay) to make transactions. The POS at the merchant approves the transaction, and *victim is charged for the transaction amount despite the card being locked by the victim*. We confirm that these flaws persist for different amounts ranging from as low as less than $1 to amounts greater than **$500** (see Table 4). We repeated these experiments several times and waited up to one week after the card was locked. However, the results of the attack remain the same.

Root causes. In-device verification comes with an assumption that the legitimate user has added the card to their wallet and that the cardholder and device owner are the same person. This leads the bank to use wallet ID and user ID interchangeably. As a result, the bank eliminates any user- and card-specific verification methods that would not be accessible to an unauthorized user. In contrast, physical cards require a PIN and signature for cardholder verification.

Also, the bank does not add an extra layer of security by allowing the POS terminal to verify the card owner before authorizing the payment. In consequence, `'No CVM required'` translates to `'successful'` verification when payments are made through a digital wallet (refer to Section 6.3.4 in EMVCo. Book4 [25] for more details).

Lessons learned. Performing the cardholder verification through device verification is both convenient and easy to implement. However, it is based on the flawed assumption that the device owner and the cardholder are the same person. The attacker can charge the victim's card using their own wallet and bypass the cardholder verification required for payment authorization.

4.3 Bypassing Access Control Policies through Recurring Transactions

Thus far we have discussed the weaknesses in authentication and cardholder verification for one-time transactions over digital wallets. Our study also finds flaws in another type of transaction: recurring payments made for subscription-based services (e.g., Netflix service). They differ from one-time transactions (e.g., in-store purchases) in three distinct ways: (i) they are initiated by the merchant, (ii) they are performed periodically, and (iii) the transaction amount is the same for every periodic transaction.

The bank treats recurring payments differently from the one-time transaction; mainly because the cardholder is penalized by the merchant for missed payments. It aims to shield the cardholder from late payments, and therefore processes the recurring payments from the wallet despite the card being locked/lost[7]. The special treatment of recurring transactions gives birth to a flaw where the attacker makes a one-time transaction on the locked card but gets it labeled as a recurring payment by the merchant. In this way, they can bypass the restrictions of one-time payment on a locked/lost card (as some banks have enforced, refer to Table 1).

Issue 4: The attacker can deceive the merchant into labeling a one-time transaction as "recurring", and bypass the access control restrictions at the bank.

Attack details. The procedure starts with the user who sets up the recurring payment contract with a merchant through the digital wallet. The wallet, acting as an agent, sends the contract details (e.g., periodicity of the payment), and the merchant information to the bank.

On receiving the request, the bank verifies the wallet and issues a merchant token `merchantID` and the `contractID` to

[7]The Citi bank states: "Any charge marked by the merchant as 'recurring', will continue to be processed while your card is locked."

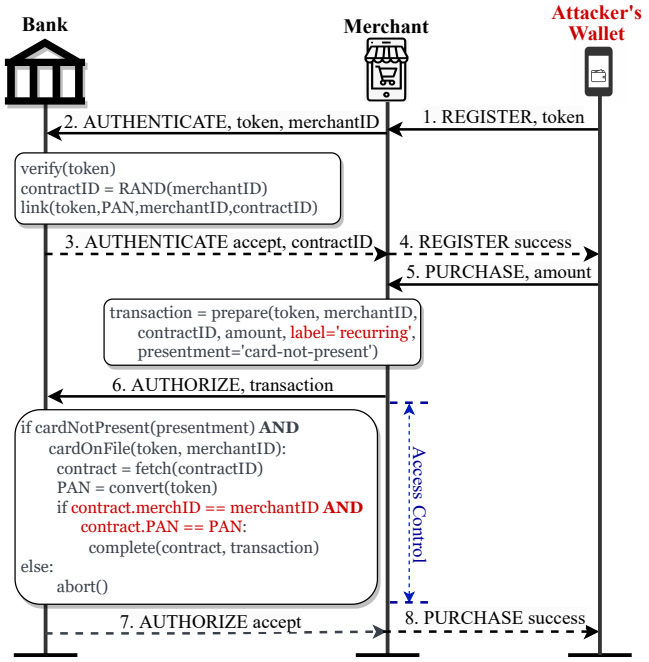


Figure 7: Access control procedure in the bank for recurring transactions. Conditions highlighted in red show the access control policy rules that allow the attacker to exploit the vulnerabilities. For example, the bank only checks the transaction label. This allows the bypass of card lock restrictions by labeling the one-time transaction(s) as recurring.

the merchant. In this way, the bank establishes trust with the merchant by giving it the authority to post recurring transactions. The bank does not perform further access control checks; instead, it processes the payments initiated for the registered `merchantID` and `contractID`.

We analyze how the bank processes recurring transactions by looking closely into the involved procedures (see Figure 7). First, the attacker registers the stolen card with an online merchant by using their digital wallet. The wallet sends a `REGISTER` request to the merchant along with its token (①). Once `REGISTER` request is received, the merchant sends the `AUTHENTICATE` request to the bank along with its `merchantID` and wallet's token (②). The bank verifies the token and sets up a recurring payment contract with the merchant to support future transactions. The bank generates a `contractID` and shares it with the merchant in `AUTHENTICATE` *`accept`* message (③). Upon receiving the `contractID`, the merchant sends a `REGISTER` *`success`* message to the wallet (④) to finalize the registration.

Merchants treat the transactions originating from the customer's saved payment method as recurring [63]. Although it is a one-time transaction, the merchant labels it as 'recurring'. The attacker uses this flaw to bypass the restrictions on one-time transactions if the card is locked. To complete the purchase, the merchant sends the `AUTHORIZE` request to the bank (⑥) along with the prepared transaction details.

Before authorizing recurring transactions, the bank's access control module verifies `card-not-present` and `card-on-file` flags to ensure that the transaction is initiated by the merchant on behalf of the user. The bank retrieves the PAN and payments contract for the merchant and verifies the transaction with the agreed-upon contract. Finally, the bank completes the authorization and sends the `AUTHORIZE` *`accept`* message to the merchant (⑦). Similarly, the merchant sends a confirmation to the wallet with `PURCHASE` *`success`* message (⑧).

Key takeaways. The bank's decision to freely allow merchants to label any transaction as 'recurring' is a trade-off between usability and security. While this design allows the users to continue using the services after they have locked the physical card, it gives birth to access control bypass vulnerability even for one-time transactions. The bank relies on its security modules (e.g., access control, and fraud detection tools) to filter out unexpected transactions by the user. However, these tools cannot differentiate between recurring and one-time transactions (especially, when the transaction is already registered at the bank); and fail to verify the nature of the transactions. In consequence, the attacker can bypass the access control policies for one-time transactions, and *can make purchases of any amount and at any time despite the card being locked by the cardholder*.

A curious reader should ask the question: how are the recurring transactions through a wallet different from those of a physical card? We should emphasize that the attacker can set up *new and future* recurring payments using their wallet even *after* the victim has locked or replaced the card. In the case of a physical card, the recurring payments set up prior to the card lock are only processed, and the attacker *cannot* set up new recurring transactions once the card is locked. The bank's indefinite and unconditional trust in the wallet opens a trap door to bypass access control policies on locked cards.

Empirical assessment. We demonstrate another way of making one-time transactions on a locked card. The attacker can bypass the access control restrictions placed by the locked card. For our experiments, we use Citibank Mastercard issued to the victim (recall that Citibank does not allow one-time transactions on locked cards 1). The attacker steals the card and adds it to their wallet *before* the victim locks the card.

The attacker visits Turo.com to make the payment for the car rental. They first register at Turo.com as a new customer and set their wallet as a payment method. Turo sends the payment method details to the card issuer bank to get the user (attacker) registered and set up a recurring transaction. After registering the card, the attacker books their car rental trip and pays in full by using the victim's card added to their wallet. Although the card is locked, Turo sends the one-time payment

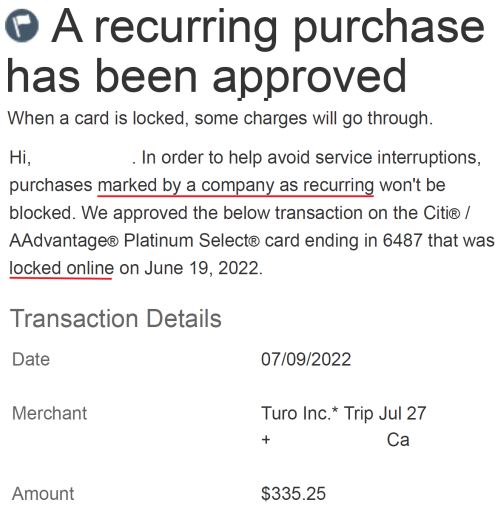


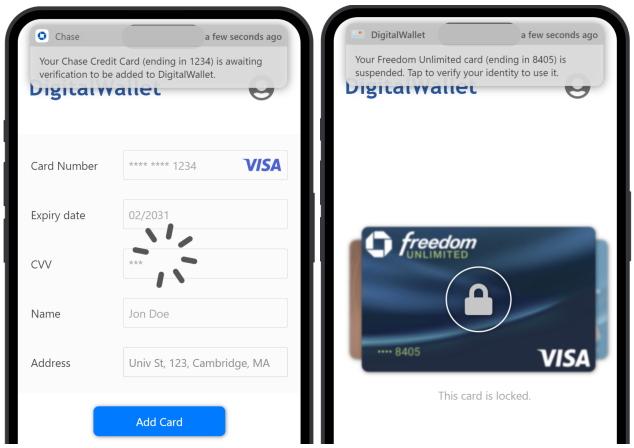
Figure 8: The bank allows the transaction by treating it as recurring without verifying the nature of the service(s) purchased. The attacker chooses to use a merchant (e.g., Turo) that labels a one-time car rental transaction as recurring to bypass the restrictions on a locked card.

by labeling it as recurring which we have confirmed from the email receipt received by the bank (refer to Figure 8).

To confirm that this vulnerability is not associated with Turo only, we also validate our attack on apple.com. Following, the above experiment steps, we have successfully purchased a $25 Apple gift card, and $179 AirPods by using the locked card. The bank's email receipt shows a recurring transaction on the card. We conclude that the bank does not verify the metadata in the recurring transactions as part of the access control policy; a recurring label is sufficient for the bank to authorize the transactions.

Root causes. The bank's access control policy is not designed to restrict the registered payment types (i.e., recurring transactions in our study). These transactions are fully trusted and not validated against the recurring payment contract (e.g., transaction amount, and periodicity). Further, the bank does not distinguish the service and product types to counter-verify the label assigned by the merchant. In consequence, those payments that are by definition one-time transactions (purchasing a gift card, and renting a car) can bypass access control restrictions when labeled as recurring.

Our analysis also reveals that the time of recurring transactions is not verified against the time when the victim has locked the card. The recurring transactions are authorized even if they are set up **after** the card loss event. The bank's access control fails to block them where it protects the edge use case scenarios: the card is locked while the recurring payment is being set up. We argue that this design choice welcomes access control violation vulnerabilities.



(a) Push MFA Authentication (b) Re-authentication on Card Lock

Figure 9: The visual representation of proposed countermeasures: (a) push MFA instead of traditional OTP-based authentication, and (b) re-authentication after card lock.

Lessons learned. The bank permits recurring payments on the locked card to honor the contract between the user and the merchant. It allows subscription-based service continuity by making timely payments on behalf of the user. The special treatment of one type of payment over the other is vulnerable to security attacks. The attacker can deceive the merchant for their one-time payment to be labeled as recurring and bypass the access control restrictions at the bank.

5 Proposed Countermeasures

We propose recommended solutions to fix the four issues that we identify in this paper. Our solutions aim to fix the fundamental design flaws and the bank's policy implementation to address the discussed issues, as well as similar others. We have suggested these solutions to banks and wallets (as noted in Table 2), but have not implemented them at our end mainly because of propriety wallet apps and inaccessible bank back-end system. Nevertheless, we have graphically illustrated in Figure 9 regarding how our solutions will look like, if implemented.

Adopting push MFA instead of traditional OTP-based methods. The existing authentication methods rely on legacy solutions that are proven to be vulnerable [42, 43]. We propose that the bank should enforce state-of-the-art MFA mechanisms [22, 52] such as push notifications (e.g., Bank App, Duo Mobile, Microsoft Authenticator), or passcodes (e.g., Google Authenticator) based solutions (see Figure 9 (a)). We propose that the bank should choose the strongest possible authentication method that it can support, rather than relying on the wallet's capability and preference. Similarly,

the wallet should prioritize security over user convenience. Our solution addresses *Issue 1*, and the vulnerabilities associated with precondition of *Issue 2* and *Issue 3* where the attacker is required to add the victim's card into their wallet. This solution requires an integration with the third party authenticator service providers, such as Duo, Google Authenticator. and Microsoft Authenticator.

Using continuous authentication in token management. Once the wallet is authenticated and a payment token is issued, the bank uses the token indefinitely which never expires. This establishes an unconditional trust towards the wallet which neither expires nor changes even for critical events like card loss, device loss, and card deletion. We propose that the banks adopt a continuous authentication protocol [10, 30, 31, 37], and re-authenticate the wallet and refresh the token periodically, especially after critical events like card loss (see Figure 9 (b)). This solution eliminates *Issue 2* and *Issue 4* directly, and *Issue 3* indirectly. Since the use of a wallet by the attacker is only possible under an unconditional trust policy, re-authentication will stop the compromised wallets (*Issue 3*) from in-store payments. As this solution involves periodic updates using a pre-existing authentication procedure, it does not require significant changes at the banking back-end.

Distinguishing one-time from recurring transactions. Currently, the bank relies on the merchant-assigned transaction type label to decide the authorization mechanism. We argue that the bank should evaluate the transaction metadata (e.g., time, frequency, and service/product type), and the transaction history information to evaluate if the transaction is of a certain type or not. For instance, a transaction labeled as recurring cannot be used to purchase a gift card. Similarly, recurring transactions whose payment amount differs from the recurring payment contract and previous payments should not be authorized. Our proposed solution resolves *Issue 4*, preventing merchants from exploiting incorrect transaction labels to circumvent access control restrictions. It can be implemented swiftly with minimal deployment efforts by amending the bank's transaction authorization logic.

6 Related Work

EMV and mobile payment security. EMV, a globally used secure payment technology, has been subject to multiple attacks. [12] identifies terminal-level vulnerabilities, leading to mismatches between card brands and payment networks, and PIN bypass. [65] demonstrates relay attacks on Dutch bank cards with minimal resources. [23] demonstrates attacks on EMV contactless cards approving unlimited transactions in foreign currencies without the cardholder's PIN. [8] introduces an "over-the-counter payment frauds" attack that exploits payment scheme designs. [2] explores the discrepancies between EMV and ISO standards, compromising transaction integrity. [54] demonstrates practical relay attacks for enabling wireless money pick-pocketing between contactless EMV bank cards and shop readers. Formal models and fine-grained analyses of EMV protocols have revealed flaws on both the cardholder and merchant sides [11].

Payment apps security. In a study of the Unified Payments Interface (UPI), [41] unveils vulnerabilities in multi-factor authentication that could lead to severe attacks when exploited by malicious applications. [45] introduces AARDroid, a tool for statically assessing payment software development kits against industry security standards. [71] highlights security weaknesses in in-app purchasing (IAP) methods within mobile games, revealing misplaced trust in payment verification or a complete lack of verification. [68] build mobile checkouts inspired by PayPal and Visa Checkout SDKs, checking payment procedure correctness and security through automatically generated test cases. [70] presents Payment-Guard, which characterizes and detects suspicious transactions based on various behavioral aspects. [40] introduces CrySL, a tool analyzing over 10,000 Android apps, uncovering widespread misuse of cryptographic APIs and vulnerabilities within their payment processes.

Security against card skimmers. Recent research highlights the prevalence of credit card skimming attacks at gas stations, with studies by [14] and [58] focusing on using smartphone Bluetooth scanning to detect skimmers. [59] introduces Skim Reaper, a tool utilizing the physical characteristics and constraints of card skimming devices for detection. [56] exposes over 50 websites hosting payment card skimming codes to steal credit card credentials.

Our research stands apart from the aforementioned studies in three key ways. Firstly, our attacks on digital wallet payment systems are novel and do not rely on preexisting vulnerabilities, e.g., in payment protocol (e.g., EVM), or digital payment elements (e.g., POS). Secondly, our attacks persist even when the victim and the bank have implemented countermeasures like card locks or the issuance of new cards with different PANs. Thirdly, our passive attacker exploits the trust among different elements in the digital wallet system to launch the attack. For instance, they can self-authenticate within the wallet to use the victim's card for transactions.

7 Conclusion

Our study reveals critical security vulnerabilities within the digital wallet payment ecosystem. These vulnerabilities are fundamentally linked to the decentralized delegation of authority. For instance, after authentication, the bank issues a payment authorization token to the wallet. The inherent weakness lies in the lack of revocation of this authority in the event of card or device loss. We demonstrate our attacks over

major US banks, popular wallet apps, and smartphone operating systems. Our findings reveal that existing remedies like card locking or issuing new cards do not effectively mitigate these threats. Finally, we propose countermeasures of strong authentication and robust token management lifecycles for digital wallets.

Acknowledgement

We thank the USENIX Security reviewers and our shepherd for their valuable feedback. This work is partially supported by NSF under grants 2345563, 1941583 (NSF Engineering Research Center for Quantum Networks), 2145631, 2215017, and 2226447, the Defense Advanced Research Projects Agency (DARPA) under contract number D22AP00148, and the NSF and Office of the Under Secretary of Defense– Research and Engineering, ITE 2326898, as part of the NSF Convergence Accelerator Track G: Securely Operating Through 5G Infrastructure Program.

References

[1] Svetlana Abramova and Rainer Böhme. Anatomy of a High-Profile data breach: Dissecting the aftermath of a Crypto-Wallet case. In *32nd USENIX Security Symposium*, pages 715–732, 2023.

[2] Nicholas Akinyokun and Vanessa Teague. Security and privacy implications of nfc-enabled contactless payment systems. In *Proceedings of the 12th international conference on Availability, reliability, and Security*, pages 1–10, 2017.

[3] Hosam Alhakami et al. Knowledge based authentication techniques and challenges. *International Journal of Advanced Computer Science and Applications*, 11(2), 2020.

[4] Reem AlHusain and Ali Alkhalifah. Evaluating fallback authentication research: a systematic literature review. *Computers & Security*, 111:102487, 2021.

[5] AMEX. Apple Pay® - frequently asked questions. https://www.americanexpress.com/us/credit-cards/features-benefits/digital-wallets/apple-pay/frequently-asked-questions.html.

[6] Apple. Apple platform security. https://help.apple.com/pdf/security/en_US/apple-platform-security-guide.pdf, May 2022.

[7] Nadarajah Asokan, Phillipe A Janson, Michael Steiner, and Michael Waidner. The state of the art in electronic payment systems. *Computer*, 30(9):28–35, 1997.

[8] Xiaolong Bai, Zhe Zhou, XiaoFeng Wang, Zhou Li, Xianghang Mi, Nan Zhang, Tongxin Li, Shi-Min Hu, and Kehuan Zhang. Picking up my tab: Understanding and mitigating synchronized token lifting and spending in mobile payment. In *USENIX Security Symposium*, pages 593–608, 2017.

[9] Dirk Balfanz and Edward W Felten. Hand-Held computers can be better smart cards. In *8th USENIX Security Symposium*, 1999.

[10] Okan Engin Basar, Gulfem Alptekin, Hasan Can Volaka, Mustafa Isbilen, and Ozlem Durmaz Incel. Resource usage analysis of a mobile banking application using sensor-and-touchscreen-based continuous authentication. *Procedia Computer Science*, 155:185–192, 2019.

[11] David Basin, Ralf Sasse, and Jorge Toro-Pozo. The EMV standard: Break, fix, verify. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 1766–1781. IEEE, 2021.

[12] David Basin, Ralf Sasse, and Jorge Luis Toro Pozo. Card brand mixup attack: Bypassing the pin in non-visa cards by using them for visa transactions. In *Proceedings of the 30th USENIX Security Symposium*, pages 179–194. USENIX Association, 2021.

[13] David Basin, Patrick Schaller, and Jorge Toro-Pozo. Inducing authentication failures to bypass credit card pins. In *32rd USENIX Security Symposium*, 2023.

[14] Nishant Bhaskar, Maxwell Bland, Kirill Levchenko, and Aaron Schulman. Please pay inside: Evaluating bluetooth-based detection of gas pump skimmers. In *USENIX Security Symposium*, pages 373–388, 2019.

[15] Maia J Boyd, Jamar L Sullivan Jr, Marshini Chetty, and Blase Ur. Understanding the security and privacy advice given to black lives matter protesters. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–18, 2021.

[16] Chase. AVS response codes. https://merchantservices.chase.com/support/faqs/address-verification-service.

[17] Ye Chen and Divakaran Liginlal. An empirical investigation of knowledge-based authentication. 2007.

[18] Penny Crosman. The case for knowledge-based authentication. *American Banker*, 1(58), 2016.

[19] Cybersource. AVS codes. https://developer.cybersource.com/library/documentation/dev_guides/Secure_Acceptance_Hosted_Checkout/html/Topics/AVS_Codes.htm.

[20] Cybersource. Global fraud and payments report. https://www.emvco.com/wp-content/uploads/2017/04/EMV_v4.3_Book_3_Application_Specification_20120607062110791.pdf, 2022.

[21] Alaa Daffalla, Lucy Simko, Tadayoshi Kohno, and Alexandru G Bardas. Defensive technology use by political activists during the sudanese revolution. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 372–390. IEEE, 2021.

[22] Dipankar Dasgupta, Arunava Roy, Abhijit Nag, Dipankar Dasgupta, Arunava Roy, and Abhijit Nag. Multi-factor authentication: More secure approach towards authenticating individuals. *Advances in User Authentication*, pages 185–233, 2017.

[23] Martin Emms, Budi Arief, Leo Freitas, Joseph Hannon, and Aad van Moorsel. Harvesting high value foreign currency transactions from emv contactless credit cards without the pin. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 716–726, 2014.

[24] EMVCo. EMV integrated circuit card specifications for payment systems – application specification. https://www.emvco.com/wp-content/uploads/2017/04/EMV_v4.3_Book_3_Application_Specification_20120607062110791.pdf, November 2011.

[25] EMVCo. EMV integrated circuit card specifications for payment systems – cardholder, attendant, and acquirer interface requirements. https://www.emvco.com/wp-content/uploads/2017/05/EMV_v4.3_Book_4_Other_Interfaces_20120607062305603.pdf, November 2011.

[26] EMVCo. Contactless specifications for payment systems – kernel 2 specification. https://www.emvco.com/terms-of-use/?u=/wp-content/uploads/documents/C-2-Kernel-2-v2.10.pdf, March 2020.

[27] EMVCo. EMV payment tokenisation specification – technical framework. https://www.emvco.com/terms-of-use/?u=wp-content/uploads/documents/EMVCo-Payment-Tokenisation-Specification-Technical-Framework-v2.3.pdf, April 2020.

[28] EMVCo. EMV payment tokenisation – a guide to use cases. https://www.emvco.com/terms-of-use/?u=wp-content/uploads/documents/EMVCo-Payment-Tokenisation-A-Guide-To-Use-Cases-v2.1.pdf, August 2021.

[29] Antonio M Espinoza, William J Tolley, Jedidiah R Crandall, Masashi Crete-Nishihata, and Andrew Hilts. Alice and bob, who the FOCI are they?: Analysis of end-to-end encryption in the LINE messaging application. In *7th USENIX Workshop on Free and Open Communications on the Internet (FOCI 17)*, 2017.

[30] Priscila Morais Argôlo Bonfim Estrela, Robson de Oliveira Albuquerque, Dino Macedo Amaral, William Ferreira Giozza, and Rafael Timóteo de Sousa Júnior. A framework for continuous authentication based on touch dynamics biometrics for mobile banking applications. *Sensors*, 21(12):4212, 2021.

[31] Priscila Morais Argôlo Bonfim Estrela, Robson de Oliveira Albuquerque, Dino Macedo Amaral, William Ferreira Giozza, Georges Daniel Amvame Nze, and Fábio Lucio Lopes de Mendonça. Biotouch: a framework based on behavioral biometrics and location for continuous authentication on mobile banking applications. In *2020 15th Iberian Conference on Information Systems and Technologies (CISTI)*, pages 1–6. IEEE, 2020.

[32] US Payments Forum. EMV payment tokenization primer and lessons learned. https://www.uspaymentsforum.org/wp-content/uploads/2019/06/EMV-Payment-Tokenization-Primer-Lessons-Learned-FINAL-June-2019.pdf, June 2019.

[33] Megan Gates. Frankenstein fraud: How synthetic identities became the fastest-growing fraud trend. *ASIS International*, May 2021.

[34] Kotie Geldenhuys. Synthetic identity fraud comes at a cost for financial institutions. *Servamus Community-based Safety and Security Magazine*, 112(2):24–28, 2019.

[35] Bill Hardekopf. Who pays for fraudulent credit card transactions? *Forbes*, November 2022.

[36] U.S. Immigration and Customs Enforcement. Hsi investigates synthetic identities scheme that defrauded banks nearly $2m. September 2022.

[37] Özlem Durmaz Incel, Seçil Günay, Yasemin Akan, Yunus Barlas, Okan Engin Basar, Gülfem Isiklar Alptekin, and Mustafa Isbilen. Dakota: sensor and touch screen-based continuous authentication on a mobile banking application. *IEEE Access*, 9:38943–38960, 2021.

[38] Drazen Jorgic. Special report: Burner phones and banking apps - meet the chinese 'brokers' laundering mexican drug money. *Reuters*, December 2020.

[39] Gregory Karp. Who pays when merchants are victims of credit card fraud? *Nerd Wallet*, May 2023.

[40] Stefan Krüger, Johannes Späth, Karim Ali, Eric Bodden, and Mira Mezini. Crysl: An extensible approach to validating the correct usage of cryptographic apis. *IEEE Transactions on Software Engineering*, 47(11):2382–2400, 2019.

[41] Renuka Kumar, Sreesh Kishore, Hao Lu, and Atul Prakash. Security analysis of unified payments interface and payment apps in india. In *29th USENIX Security Symposium*, pages 1499–1516, 2020.

[42] Kevin Lee, Benjamin Kaiser, Jonathan Mayer, and Arvind Narayanan. An empirical study of wireless carrier authentication for sim swaps. In *Sixteenth Symposium on Usable Privacy and Security*, 2020.

[43] Zeyu Lei, Yuhong Nan, Yanick Fratantonio, and Antonio Bianchi. On the insecurity of sms one-time password messages against local attackers in modern mobile devices. In *Network and Distributed Systems Security (NDSS) Symposium*, 2021.

[44] Jiadong Lou, Xu Yuan, and Ning Zhang. Messy states of wiring: Vulnerabilities in emerging personal payment systems. In *30th USENIX Security Symposium*, pages 3273–3289, 2021.

[45] Samin Yaseer Mahmud, K Virgil English, Seaver Thorn, William Enck, Adam Oest, and Muhammad Saad. Analysis of payment service provider sdks in android. In *Proceedings of the 38th Annual Computer Security Applications Conference*, pages 576–590, 2022.

[46] Nicholas R Merker, Nicole R Woods, and Blaine L Dirker. The password is dead; is knowledge-based authentication far behind. *Banking LJ*, 134:375, 2017.

[47] Stig Frode Mjølsnes and Chunming Rong. On-line e-wallet system with decentralized credential keepers. *Mobile Networks and Applications*, 8:87–99, 2003.

[48] Sila Money. The future is now: How digital wallets are transforming fintech.

[49] FDIC Consumer News. A closer look at mobile banking: More uses, more users. *FDIC*.

[50] Yellow Pages. White pages - people search and find phone numbers. https://people.yellowpages.com/whitepages/.

[51] PayPal. AVS, CVV2, and payment advice response codes. https://developer.paypal.com/api/nvp-soap/AVSResponseCodes/.

[52] Philip Polleit and Michael Spreitzenbarth. Defeating the secrets of otp apps. In *2018 11th International Conference on IT Security Incident Management & IT Forensics (IMF)*, pages 76–88. IEEE, 2018.

[53] Ariel Rabkin. Personal knowledge questions for fallback authentication: Security questions in the era of facebook. In *Proceedings of the 4th Symposium on Usable Privacy and Security*, pages 13–23, 2008.

[54] Andreea-Ina Radu, Tom Chothia, Christopher JP Newton, Ioana Boureanu, and Liqun Chen. Practical EMV relay protection. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1737–1756. IEEE, 2022.

[55] Juniper Research. Over 60% of global population to use digital wallets in 2026. https://www.juniperresearch.com/press/digital-wallet-users-exceed-5bn-globally-2026, 2022. Accessed: 2023.

[56] Phoebe Rouge, Christina Yeung, Daniel Salsburg, and Joseph A Calandrino. Checkout checkup: Misuse of payment data from web skimming, 2020.

[57] Kevin A Roundy, Paula Barmaimon Mendelberg, Nicola Dell, Damon McCoy, Daniel Nissani, Thomas Ristenpart, and Acar Tamersoy. The many kinds of creepware used for interpersonal attacks. In *2020 IEEE Symposium on Security and Privacy (SP)*, pages 626–643. IEEE, 2020.

[58] Nolen Scaife, Jasmine Bowers, Christian Peeters, Grant Hernandez, Imani N Sherman, Patrick Traynor, and Lisa Anthony. Kiss from a rogue: Evaluating detectability of pay-at-the-pump card skimmers. In *2019 IEEE Symposium on Security and Privacy (SP)*, pages 1000–1014. IEEE, 2019.

[59] Nolen Scaife, Christian Peeters, and Patrick Traynor. Fear the reaper: Characterization and fast detection of card skimmers. In *27th USENIX Security Symposium*, pages 1–14, 2018.

[60] SearchBug. Searchbug people finder. https://www.searchbug.com/.

[61] Scott Shane and Colin Moynihan. Drug agents use vast phone trove, eclipsing N.S.A.'s. *The New York Times*, September 2013.

[62] Ben Smyth. Forgotten your responsibilities? how password recovery threatens banking security. Technical report, Citeseer, 2010.

[63] Stripe. Recurring payments. https://stripe.com/docs/recurring-payments.

[64] Acams Today. The nature of synthetic identity fraud. December 2022.

[65] Jordi van den Breekel and B Asia. Relaying EMV contactless transactions using off-the-shelf android devices. *BlackHat Asia, Singapore*, 2015.

[66] Visa. Leading the token transformation. https://usa.visa.com/partner-with-us/payment-technology/visa-tokenization.html.

[67] David S Wall. Micro-frauds: virtual robberies, stings and scams in the information age. In *Corporate hacking and technology-driven crime: Social dynamics and implications*, pages 68–86. IGI Global, 2011.

[68] Quanqi Ye, Guangdong Bai, Naipeng Dong, and Jin Song Dong. Inferring implicit assumptions and correct usage of mobile payment protocols. In *Security and Privacy in Communication Networks: 13th International Conference, SecureComm 2017, Niagara Falls, ON, Canada, October 22–25, 2017, Proceedings 13*, pages 469–488. Springer, 2018.

[69] Fernando Zandona. Cash is no longer king: Why the rise of digital wallets will shift the payment landscape. *Financial IT*, February 2022.

[70] Yadong Zhou, Tianyi Yue, Xiaoming Liu, Chao Shen, Lingling Tong, and Zhihao Ding. Payment-guard: Detecting fraudulent in-app purchases in ios system. *Neurocomputing*, 422:263–276, 2021.

[71] Chaoshun Zuo and Zhiqiang Lin. Playing without paying: Detecting vulnerable payment verification in native binaries of unity mobile games. In *31st USENIX Security Symposium*, pages 3093–3110, 2022.