No Wallet for Old Tricks: Introducing Context-aware Defenses Against Cryptocurrency-based Social Engineering Attacks

Muhammad Muzammil
Stony Brook University
New York, USA
mmuzammil@cs.stonybrook.edu

Harsh Shah
Stony Brook University
New York, USA
harsh.h.shah@proton.me

Nick Nikiforakis
Stony Brook University
New York, USA
nick@cs.stonybrook.edu

Abstract

Cryptocurrency users routinely fall victim to social engineering attacks. Attackers trick victims into mistakenly authorizing transactions that transfer funds to attacker-controlled wallets rather than the intended recipient. Existing protections against these attacks largely rely on public blocklists, which attackers can easily evade. Official reports and recent measurement studies indicate that such attacks have led to billions of dollars in financial losses that are irreversible due to the immutable nature of blockchain transactions. In this work, we introduce the idea that cryptocurrency wallets can incorporate an additional context-aware, client-side layer of defense. Using this layer, wallets can better protect users from social engineering attacks by tailoring their defense to user's specific cryptocurrency activity, instead of exclusively relying on global blocklists. To demonstrate this approach, we design and implement detection mechanisms targeting three prevalent attack vectors in the most widely used Ethereum wallet, MetaMask. We show that these mechanisms can be integrated without substantial engineering complexity and incur negligible runtime overhead. We hope that our work encourages wallet providers to consider incorporating them into their existing security workflows.

CCS Concepts

• **Security and privacy → Domain-specific security and privacy architectures**.

Keywords

Typosquatting, Dropcatching, Address Poisoning, Ethereum

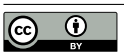
ACM Reference Format:
Muhammad Muzammil, Harsh Shah, and Nick Nikiforakis. 2026. No Wallet for Old Tricks: Introducing Context-aware Defenses Against Cryptocurrency-based Social Engineering Attacks. In *19th European Workshop on Systems Security (EuroSec '26), April 27–30, 2026, Edinburgh, Scotland Uk*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3803525.3804977

1 Introduction

As modern systems continue to harden and technical vulnerabilities become increasingly difficult for attackers to exploit, compromising infrastructure through purely technical means has become substantially more challenging. Consequently, adversaries continue to rely on social engineering strategies that manipulate users into performing actions that undermine their own security [56]. Over the past two decades, users on the web have faced a broad spectrum of such attacks that have also been formally recognized by governments and national cybersecurity agencies worldwide [5, 10, 53]. In response, defenders have deployed a variety of mitigations designed to help users identify and avoid these threats. Modern web browsers and email providers, for example, integrate phishing- and typosquatting-detection mechanisms that warn users when they attempt to access known malicious domains [23].

More recently, cryptocurrencies have emerged as a lucrative medium for social-engineering attacks. Cryptocurrencies are digital assets secured by blockchain technology, enabling peer-to-peer transactions without the need for intermediaries, such as, banks or payment processors. Each transaction is permanently recorded on an immutable and decentralized ledger, making it irreversible. Attackers frequently exploit this property by deceiving users into sending funds to addresses under their control. In contrast to traditional web services where users may recover compromised accounts or reverse fraudulent transactions, victims in the cryptocurrency ecosystem typically have no recourse once a malicious transfer has been confirmed on the blockchain. In 2024, the US Internet Crime Complaint Center reported a total of 9.3 billion USD in losses to cryptocurrency-related fraud, a large share of which is attributed to social engineering attacks [21].

Several forms of social engineering targeting cryptocurrency users have attracted significant attention due to their prevalence and financial impact [2, 24, 25, 29, 37–39, 47, 49, 50, 64]. One prominent example is *address poisoning*, in which attackers generate addresses that closely resemble those appearing in a victim's transaction history and insert them via small transfers [8, 13, 24, 25, 64, 68]. Because wallets prominently display recently used addresses, victims may mistakenly select or copy an attacker-controlled lookalike when initiating a transaction, resulting in irreversible on-chain losses (Section 3.1). Public reports document a single address-poisoning incident in 2024 resulting in a loss of $68.7M [8, 55], while measurement studies estimate cumulative losses in the hundreds of millions of dollars over a two-year period [24, 64]. To mitigate such risks, users are often advised to rely on blockchain-based naming services such as the Ethereum Name Service (ENS) [9, 17], which map human-readable names (e.g., `johndoe.eth`) to cryptocurrency addresses and reduce reliance on long hexadecimal strings. However, this usability improvement introduces security risks analogous to those observed in DNS, often with more severe financial consequences. Attackers register typographic variants of legitimate ENS names to exploit user input errors, or monitor and re-register



EuroSec '26, Edinburgh, Scotland Uk

ACM ISBN 979-8-4007-2603-3/2026/04
https://doi.org/10.1145/3803525.3804977

expired ENS domains to intercept transactions intended for the original owner (Section 3.2). Both these attacks have led to substantial financial losses for ENS domain owners [49, 50, 62, 67].

Prior work has shown that cryptocurrency wallets provide little to no effective protection against social engineering attacks. Guan and Li [25] report that none of 53 popular Ethereum wallets meaningfully defend against address-poisoning attacks, while Muzammil *et al.* [49, 50] find that widely deployed wallets lack defenses against domain-based attacks targeting blockchain naming services. Instead, existing wallet protections primarily rely on third-party blocklists [12, 41–43].

For instance, MetaMask collaborates with a third-party service that detects and maintains feeds of known malicious addresses [42]. When a user interacts with an address appearing in these feeds, MetaMask displays a warning in the wallet interface. However, to preserve user privacy, MetaMask does not share transaction details with these providers. Consequently, these services can only flag addresses that have already been identified as malicious and cannot assess whether a user is vulnerable in a given transaction involving a previously unseen address. Such approaches do provide users with protection against known malicious addresses but are inherently insufficient by themselves, as social engineering attacks are often highly targeted and unlikely to be captured promptly by public threat feeds. As a result, users remain vulnerable precisely at the moment they authorize transactions.

In this paper, we build on this insight and advance the idea that cryptocurrency wallets can deploy client-side, context-aware defenses against social engineering attacks, complementing existing protections that primarily rely on public blocklists. Such defenses can leverage wallet-specific context and user interaction patterns to detect attacks that are otherwise difficult to capture using centralized approaches. To demonstrate the feasibility of this idea in practice, we develop in-wallet defenses against the aforementioned attacks by extending the MetaMask browser extension [40], which is currently the most widely used Ethereum wallet [11]. We detect these attacks before a transaction is broadcasted to the blockchain, providing users with an opportunity to protect their funds before they are irreversibly transferred to an attacker. In our extended version of MetaMask, when a user enters a *poisoned* address or suspicious ENS name, the wallet displays an inline warning directly within the transaction-confirmation interface, informing the user of the specific risk associated with the recipient. These warnings appear at the moment the user reviews the transaction details and require explicit acknowledgement before the user may proceed.

Our detection mechanisms required only 1,356 lines of code, accounting for 0.21% of the MetaMask codebase by line count. The implementation is publicly available at [48]. This demonstrates that such defenses can be implemented through targeted and localized engineering efforts. We evaluate our implementation against confirmed address-poisoning attacks reported by Toxin Tagger [13] and find that it successfully detects *all reported cases*. We further benchmark runtime performance and observe that detection executes orders of magnitude faster than routine wallet operations, such as ENS resolution, introducing only negligible overhead. While our prototype is implemented within MetaMask, the defenses themselves are generic and can be ported to other wallets.

2 Background

Ethereum and ERC-20 Tokens. Ethereum is a public, decentralized blockchain that enables users to transfer value and interact with smart contracts without centralized intermediaries. Transactions are recorded on an immutable ledger, making transfers effectively irreversible once confirmed. Beyond native currency transfers, Ethereum supports programmable smart contracts that enable a wide range of decentralized applications [20]. Fungible assets on Ethereum commonly follow the ERC-20 token contract standard, which specifies a standardized set of functions and events (e.g., `transfer`, `approve`) that define how tokens behave and how wallets and applications should interact with them. As ERC-20 deployment is permissionless, anyone may create and distribute tokens with arbitrary properties.

Ethereum identifies entities using 20-byte addresses, displayed as 42-character hexadecimal strings prefixed with `0x`. These addresses correspond either to externally owned accounts (EOAs), which are controlled by users via private keys, or to smart contracts, which contain executable code and respond to transactions sent to them. From the user's perspective, both address types are interacted with uniformly through wallet interfaces, which prominently display truncated address representations during transaction workflows.

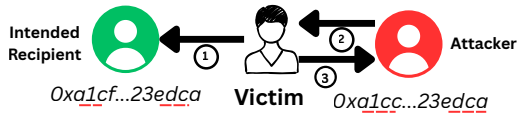
Blockchain-based Naming Services. Blockchain-based naming services map human-readable identifiers to cryptocurrency addresses, improving usability over hexadecimal strings. A prominent example is the Ethereum Name Service (ENS) [17], which is the most widely adopted. ENS domains use the `.eth` namespace, and ownership of these names is implemented using ERC-721 tokens, enabling ownership to be recorded and transferred on-chain. ENS registrations are time-limited and require periodic renewal at a price that depends on the domain's length (e.g., a 5+ character ENS domain costs $5 per year) [18]. After expiration, domains enter a grace period and may subsequently be reclaimed by new owners [19, 49, 67].

3 Threat Models

3.1 Address Poisoning

Address poisoning attacks exploit the way users often rely on their wallet's transaction history when sending funds. Many digital wallets display recent addresses that the user has previously interacted with. Because of this feature, while sending transactions to a known contact, users often select the recipient address from the wallet's history rather than manually entering the address or copying it from a trusted source. This common user behavior is exploited in address poisoning attacks, which happens in three steps.

First, the victim user makes a legitimate transaction to a trusted recipient address, which gets recorded in their wallet's recent activity list. Next, attackers generate a large number of cryptocurrency addresses that are visually similar to an address in the victim's transaction history. To do this, they repeatedly create EOAs until the hexadecimal representation of their address shares a short prefix/suffix with the target (often 5–8 hex characters total). This choice is also optimal as wallet user interfaces often truncate addresses in the middle for better readability, showing only the beginning and end characters to users. The attacker then sends a tiny-value transaction (such as 0.000001 ETH) from one of the attacker-controlled lookalike addresses to the victim. This causes the attacker's address



Victim's Transaction History

From	To	ERC-20 Token Amount	
0xa1cf...23edca		20.0	Recipient → Victim
	0xa1cf...23edca	12.3	Victim → Recipient
	0xa1cc...23edca	12.3	*Poisoning Transfer*
	0xa1cc...23edca	7.7	Victim → Attacker

Figure 1: Illustration of an address poisoning attack.

to appear in the victim's recent-activity list inside their wallet. Since many wallets automatically display or suggest recently seen addresses when the user is preparing a new transaction, this "poisons" the history by placing the attacker's lookalike address where the victim expects to find the legitimate one.

Next to tiny-value transfers, attackers have developed more sophisticated variants of this attack, which use zero-value transfers and counterfeit tokens. In the zero-value variant, the attacker invokes the `transferFrom (sender, receiver, amount)` function from the token smart contract with amount 0 (the token transfer amount) but sets the victim as sender and the lookalike as receiver in the function parameters. Since many ERC-20 implementations permit arbitrary sender/receiver for zero-value calls, the victim's history now shows an apparent prior "send" to the lookalike. In the counterfeit-token variant, the attacker deploys a fake ERC-20 token with a familiar name or symbol, then logs a transfer from the victim to the lookalike for the same nominal amount; because token names/symbols are unrestricted, near-copies are trivial, and the attacker can script transfers on behalf of any address.

Both variants increase plausibility in the victim's recent-activity list, raising the chance of a successful address poisoning attack. Later, when the user intends to send funds to the legitimate recipient, they open their wallet's recent addresses list and choose the attacker-controlled lookalike address, believing it is the one they actually intend to use as the prefix/suffix match and the interface hides the full address. The user then initiates a transfer to the attacker-controlled address, and irreversibly loses their funds to the attacker. We illustrate this attack in Figure 1.

3.2 ENS Attacks

ENS domains are often cited as a solution to the challenges posed by address poisoning attacks, as they are easier to verify visually compared to addresses. However, ENS domains suffer from most of the same vulnerabilities as traditional DNS domains, including typosquatting and dropcatching attacks, both shown in Figrue 2.

3.2.1 Typosquatting Attacks. In a typosquatting attack, adversaries register domain names that closely resemble popular legitimate ones. These attacks exploit the fact that users often mistype URLs in their browser's address bar, causing them to land on the attacker's site. Once on the attacker's page, victims may be deceived into entering sensitive information, downloading malware, or completing

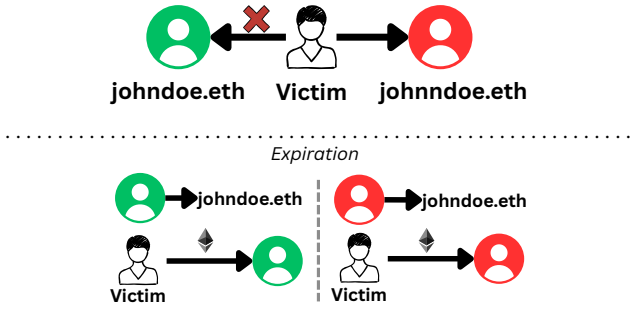


Figure 2: Illustrations of ENS-based social engineering attacks: typosquatting (top) and domain dropcatching (bottom).

fraudulent financial transactions. Contrastingly, if a user accidentally mistypes the ENS domain to the attacker controlled domian, the ENS resolver immediately returns the attacker's address and the consequences are more direct. Any funds that the user sent will now be transferred into the attacker's wallet. As Ethereum transactions are irreversible once confirmed, even a single-character mistake results in an immediate and permanent loss of funds.

3.2.2 ENS Dropcatching. Dropcatching in ENS refers to the practice of registering recently expired (or "dropped") ENS domain names that were not renewed by their previous owners. This is done so that attackers can take advantage of the residual trust and recognition associated with these domains. An ENS domain that has been in use for some time may receive funds from other users who are familiar with it but unaware that it has changed ownership, believing they are sending funds to the previous owner. According to Muzammil *et al.* [49], attackers prefer to register dropped ENS domains that were previously linked to wallets with large balances. As with typosquatting, dropcatching does not require compromising wallets or the ENS protocol, only exploiting predictable user behavior and the trust previously associated with the domain. Moreover, even if a dropcatching or typosquatting ENS domain is registered without any malicious intent, any funds mistakenly sent to it are still irretrievably lost to the new owner.

4 Attack Detection Layer

On the traditional web, responsibility for detecting and preventing social engineering attacks is largely delegated to upstream infrastructure, including web browsers, search engines, email gateways, and hosting providers. These systems automatically block known malicious domains and filter phishing content, so that the as few users as possible are exposed to them. However, in the context of cryptocurrency transactions, the digital wallet is the sole intermediary between the user and the blockchain. Unlike the traditional web, there is no browser-equivalent safety layer capable of intercepting suspicious activity before a transaction is finalized. Once a user enters an address or selects an ENS name, the wallet is the only component with the necessary visibility to detect typographical mistakes or flag malicious addresses prior to issuing a transaction on the blockchain. For this reason, we argue that integrating attack-detection mechanisms directly into the wallet offers the most effective and practical defense against social engineering attacks in the cryptocurrency ecosystem.

To implement the defenses described in Section 3, we extend the open-source MetaMask browser extension wallet, which is the most widely used Ethereum wallet at the time of writing [11]. As MetaMask presents the same flow for benign and malicious addresses alike (including ENS domains), users see no visual cue to distinguish a risky transaction from a safe one, making it harder for them to protect themselves against social engineering attacks. Figure 5 in Appendix A shows a typical Ethereum transaction flow (without any malicious activity detection) using the MetaMask wallet. Our modifications augment the transaction-confirmation interface by incorporating additional security checks and warnings that will be shown to the user when they are confirming the transaction.

We implement three detection mechanisms corresponding to the attack vectors discussed earlier, and whenever a transaction triggers any of these mechanisms, the wallet displays a prominent warning, requiring the user to explicitly acknowledge the risk before proceeding. This design is in-line with MetaMask's existing threat-warning infrastructure while extending it to cover social-engineering attacks specific to address poisoning and ENS names.

5 Detection Mechanisms

In this section, we describe the specific detection mechanisms we implement inside the MetaMask wallet to defend against the attack vectors outlined in Section 3. We note that our implementations can be adapted to other wallet software across different device types, including mobile wallets and hardware wallets with companion applications. The core detection logic remains the same, with only adjustments needed to accommodate different storage models or user interface designs. Illustrations of the detection workflows for each attack vector are shown in the Appendix (Figures 4, 7, and 6).

5.1 Address Poisoning Detection

To defend against the address poisoning attack described in Section 3.1, we implement a client-side detection mechanism inside the MetaMask wallet. The detection logic is distributed across three components: (i) A state-management layer that maintains the core detection algorithm, (ii) A set of UI hooks that expose this logic to the interface, (iii) A confirmation-flow alert system that warns users during transaction preparation. This integration allows the wallet to evaluate the safety of a recipient address as soon as it is entered in the *send* form while the user is preparing a transaction. If the address is flagged as potentially poisoned, the wallet displays an alert to the user before the transaction is finalized (see Figure 4).

The mechanism relies on two pieces of locally stored information: (i) A record of all addresses to which the user has previously sent transactions (outgoing transaction history), and (ii) A record of all addresses from which the user has previously received transactions (incoming transaction history). These two transaction histories are sufficient to capture the characteristic interaction pattern of address poisoning without requiring external services or global blocklists.

The wallet compares the entered address against recipient addresses in the user's outgoing and incoming history by matching configurable prefix/suffix characters. Prior work suggests that checking 7 characters (3 from the prefix and 4 from the suffix) balances false-positive rate with effective detection of address-poisoning attempts [64, 68]. We revisit these thresholds in Section 6. If the entered address is both unfamiliar and visually similar to a known recipient, the wallet classifies the transaction as potentially poisoned. It then displays an inline warning during the confirmation step, prompting the user to double-check the recipient before proceeding. The warning requires explicit acknowledgement from the user before proceeding with the transaction. All checks run locally and in real time.

5.2 ENS Attack Detection

5.2.1 ENS Typosquatting Detection. To mitigate the ENS typosquatting attack described in Section 3.2, we implement client-side detection inside the MetaMask wallet, following the same three-component architectural pattern as the address poisoning defense.

The typosquatting detection pipeline is triggered whenever the user enters an ENS domain in the recipient field of the send interface. Instead of immediately resolving the name to an Ethereum address, the wallet first retrieves the set of ENS domains the user has interacted with in past transactions. As MetaMask does not, by default, maintain a history of ENS domains used by the user, we extend the wallet to store this information locally in browser extension's storage (which MetaMask already uses extensively to maintain user metadata). This extension storage is a persistent key-value store that remains available across sessions and is sandboxed on a per-extension basis. Hence, only MetaMask's own scripts may access this data. After each successful transaction involving an ENS-based recipient, the domain is added to this local store.

The data is automatically synchronized across devices associated with the same browser profile. On other platforms, equivalent state can be maintained using platform-provided persistent storage. For instance, the Android application for MetaMask or any other wallet can store state in sandboxed persistent file storage [4].

Once the reference set is retrieved, the wallet evaluates whether the newly entered domain is suspiciously similar to any previously used domain. To detect ENS typosquatting, the wallet compares the entered domain against each domain in the reference set using a set of predefined typo models, including single-character substitutions (e.g., `johndoe.eth` → `johnxoe.eth`), deletions (e.g., `johndo.eth`), insertions (e.g., `johndfoe.eth`), duplications (e.g., `johnddoe.eth`), adjacent-character transpositions (e.g., `johndeo.eth`), and trailing pluralization (e.g., `johndoes.eth`). These typosquatting models have been used extensively in prior work on detecting typosquatting on both ENS and DNS domains on the traditional web and capture common typographical mistakes that users make when entering domain names [3, 6, 45, 50, 61, 63, 65]. The comparison between the entered domain and each previously used domain is performed using an edit-distance metric known as the Damerau-Levenshtein distance, which is a string similarity metric specifically designed to model the types of mistakes humans naturally make when typing [14]. It measures the minimum number of single-character operations required to convert one string into another.

If the entered domain name differs by a single edit from a previously used domain and the typo model matches one of those listed above, the wallet marks the lookup as suspicious and attaches a typosquatting warning to the resolved address. This warning is shown during the transaction confirmation step through the wallet's UI alert system. The warning does not block the transaction but prompts the user to verify the domain's correctness (Figure 7).

5.2.2 ENS Dropcatching Detection. To detect ENS dropcatching attacks, we implement a client-side mechanism in MetaMask that tracks the historical resolution of ENS domains the user has previously transacted with. Whenever the user enters an ENS name in the recipient field, the wallet resolves the domain and retrieves the Ethereum address it currently maps to. In parallel, the wallet consults a locally stored historical record of domain-to-address mappings that were saved after successful past transactions involving ENS names. This allows MetaMask to determine whether the domain currently resolves to the same address as before or whether its resolution has changed since the last interaction.

If the domain has never been used by the user, or if there is no stored history for the domain, the wallet treats the lookup as a first-time use and does not trigger any warnings (Section 6.2). However, if the domain has been used previously, the wallet compares the current resolved address with the historical address recorded at the time of the last transaction. A mismatch indicates that the domain now resolves to a different address than before, which a signal of a potential dropcatching attack. In such cases, the wallet classifies the transaction as high-risk and displays a warning in the confirmation interface informing the user that the destination address associated with the domain has changed (Figure 6).

Similar to address poisoning and typosquatting detection, the warning does not block the transaction but prompts the user to verify the legitimacy of the address change. This is particularly important because ENS ownership changes and administrative updates do occur legitimately, and the user might actually want to send the transaction to the new owner of the domain. Nevertheless, the wallet's detection mechanism ensures that the user is aware of the change before authorizing a transfer. All checks run locally and in real time, using only wallet-stored data and publicly available ENS resolution results. No external APIs, remote services, or global blocklists are required.

6 Evaluation and Discussion

6.1 Runtime Performance Analysis

In this section, we describe our experiments for evaluating the runtime performance of our detection mechanisms. High runtime overhead could negatively impact the user experience in cryptocurrency wallets, so it is important to assess whether these defenses can operate efficiently during normal wallet usage. We evaluate each attack vector separately.

For address-poisoning detection, we measure runtime across accounts with varying numbers of historical transactions, since the detection logic compares the entered recipient address against all addresses in the user's transaction history. We evaluate performance under four transaction history sizes: 20, 500, 1,000, and 10,000 transactions. Similarly, for ENS typosquatting and ENS domain dropcatching, we measure performance while varying the number of previously used ENS domains stored in the extension's local storage, as the detection mechanisms compare the entered domain against this locally maintained reference set.

All experiments were conducted using the same browser instance, with no additional extensions or tabs loaded. Each measurement was repeated three times, and we report the average runtime. Across all experiments, we observe that the detection mechanisms complete within tens of milliseconds. Dropcatching detection is the fastest, consistently executing in under 10 ms. The only notable outlier is ENS typosquatting detection with a history of 10,000 domains, which takes approximately 290 ms. These overheads remain negligible in practice, particularly given that routine wallet operations incur higher latency. For instance, resolving an ENS domain consistently takes around 150 ms.

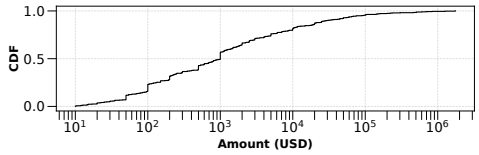


Figure 3: CDF of the total USD lost in the address-poisoning attacks reported by Toxin Tagger [13] that are successfully detected by our MetaMask in-wallet poisoning defense.

6.2 Detection Effectiveness

Generating Ethereum addresses with chosen prefixes or suffixes is computationally expensive due to the randomness of private-key generation and the lack of structural shortcuts in ECDSA-based address derivation [32]. Attackers must repeatedly generate keypairs until an address meets the desired string pattern, making the search effort grow exponentially with every additional hexadecimal character they attempt to control. Ye *et al.* [68] show that even on a high-end multi-core machine, producing an address with seven specific hex digits requires several hours of continuous computation, whereas targeting ten specific digits increases the workload to more than 600 days. In the context of address-poisoning detection, lowering the similarity threshold increases the likelihood of false positives, while raising it risks missing attacker-generated lookalikes. Prior work has shown that requiring a 3-character prefix and 4-character suffix match can capture a substantial portion of real-world poisoning attempts [64]. Building on these observations, our defense allows users to configure their preferred prefix and suffix thresholds according to their own risk tolerance. By default, we adopt the same (3, 4) split.

To evaluate the effectiveness of our address-poisoning detection implementation, we test our implementation against the dataset published by Toxin Tagger [13], a real-time feed of poisoning attacks compiled by Tsuchiya *et al.* [64]. Using the same parameters as Tsuchiya *et al.* (a match of at least 3 prefix and 4 suffix characters) we achieve 100% detection coverage over all poisoning attacks listed in the dataset. The total amount of funds lost through these 436 address poisoning transactions was 11M USD. Figure 3 displays a CDF of the losses of funds through these transactions. Increasing the similarity threshold reduces detection coverage and is therefore not recommended in practice. Owing to the high entropy of Ethereum addresses, the default (3, 4) prefix-suffix split already yields a low probability of false positives. When the threshold is raised to 5 prefix and 5 suffix characters, the detection rate drops substantially, missing roughly two-thirds of the attacks, as expected under a stricter matching requirement. While more stringent configurations, such as (4, 4), further decrease the likelihood of false positives, they do so at the cost of significantly reduced detection coverage, making them overly conservative for practical deployment.

To evaluate our methodology for detecting ENS typosquatting attacks, we collected registered typosquatting variants of ENS domains reported by Muzammil *et al.* [50]. Their study identifies typosquatting domains targeting popular cryptocurrency influencers that actively receive transactions from unsuspecting users. From this dataset, we selected typosquatting variants corresponding to each of the six typosquatting models described in Section 5.2. As a representative example, several variants such as `fitalik.eth` and `vialik.eth` target the ENS domain `vitalik.eth`, owned by the Ethereum founder Vitalik Buterin. We evaluated our implementation using these variants and observed that the system successfully detected all attack instances, provided that a transaction to the legitimate domain `vitalik.eth` had occurred prior to sending a transaction to the typosquatting variants. To evaluate ENS dropcatching detection, we registered an ENS domain and first performed a transaction through that domain. We then modified the address associated with the domain to a different address, simulating the behavior of a new owner who acquires an expired ENS name and updates its resolution. In this scenario, our implementation correctly generated a warning indicating the change in address.

6.3 Future Directions

We note that our detection methodology relies on the victim's prior interaction with a benign address or ENS domain. Consequently, additional safeguards are required in cases where a victim encounters a malicious address or domain without having previously interacted with the legitimate counterpart. Currently, wallet providers maintain blocklists of known malicious addresses and display warnings when users attempt to interact with them. However, blocklists for ENS domains are not maintained. Wallet providers could extend their existing security mechanisms to include ENS domains and further perform client-side analysis of destination addresses before a transaction is finalized. The traditional web ecosystem offers useful analogies. For example, security vendors often analyze newly registered DNS domains to identify suspicious activity associated with newly created infrastructure [27, 51, 52]. Similarly, wallets could flag addresses that are newly created or have very few transactions, as these characteristics may indicate higher risk. Last, to the extent that a list of popular and benign ENS domains can be curated, these could be preloaded in a wallet, conceptually similar to preloaded HSTS websites on the regular web [28]. In this way, even if, for instance, a user has never interacted with `vitalik.eth`, they would still see a warning if they try to send funds to `vialik.eth`.

If cryptocurrency wallets incorporate in-wallet defenses of this form, users will be alerted before completing malicious transfers resulting from attacks specifically tailored to them, thereby preventing losses such as those shown in Figure 3. More broadly, integrating such protections across the wallet ecosystem would strengthen user protection. Wallet developers could extend our approach to additional attack vectors and incorporate adaptive thresholds that evolve in response to changing attacker behavior.

7 Related Work

To the best of our knowledge, no cryptocurrency wallet currently protects their users with context-aware defenses against social engineering attacks. Guan and Li [25] evaluated 53 cryptocurrency wallets and found that only three displayed warnings for malicious addresses. None of the wallets recognized the defining characteristics of address poisoning, such as one-sided dust transfers from lookalike addresses, nor did they provide interface cues that help users distinguish suspicious inbound transfers or verify intended recipients. Tsuchiya *et al.* [64] analyzed more than two years of Ethereum and Binance Smart Chain transaction data and identified approximately 270 million poisoning attempts targeting 17 million potential victims, with 6.6 thousand successful incidents resulting in an estimated $84 million in losses.

Moreover, multiple studies of how users interact with cryptocurrency wallets have concluded that the average cryptocurrency wallet user routinely underestimates or is unaware of the security risks involved in managing digital assets [1, 2, 16, 22, 26, 31, 33, 66]. Users frequently hold incorrect mental models about how security mechanisms in wallets operate, such as how private keys are generated or secured, and how transactions should be verified. These misunderstandings ultimately manifest as a range of unsafe behaviors. These research efforts motivate the need to develop features within cryptocurrency wallets that make it easier for novice (and even experienced) cryptocurrency users to protect themselves from social engineering attacks.

Typosquatting and domain dropcatching are long-standing threats on the web and have been studied extensively in the context of domain abuse and user deception [3, 6, 34–36, 44–46, 59–61, 63, 65]. Recent work has shown that these attacks also affect blockchain-based naming services, including ENS and related systems [15, 29, 30, 49, 50, 67]. Muzammil *et al.* [50] demonstrate that attackers target both high-visibility domains (e.g., `vitalik.eth`) and domains associated with high-value blockchain activity, while follow-up work on ENS dropcatching reveals widespread domain re-registration [49]. Beyond naming-service abuse, cryptocurrency users are frequently targeted by social engineering attacks that operate outside the blockchain layer, adapting traditional web-based scam techniques such as fraudulent giveaways, investment schemes, and fake recovery services for victims of such scams [2, 7, 37–39, 49, 54, 57, 58].

8 Conclusion

In this paper, we introduced the idea of in-wallet, client-side defenses against cryptocurrency-based social engineering attacks. Through a concrete implementation in the MetaMask wallet, we demonstrated how contextual, real-time warnings can be integrated at the moment users authorize transactions, when mistakes may otherwise lead to irreversible financial loss. Our evaluation shows that these mechanisms can be incorporated without substantial engineering complexity and incur negligible runtime overhead, while effectively detecting known instances of these attacks. We hope that this work encourages wallet providers to consider adopting client-side, context-aware defenses as a complement to existing security mechanisms and to further explore their role in protecting users against social engineering attacks.

Acknowledgements. This research was developed with funding from the Defense Advanced Research Projects Agency (DARPA). Any opinions, findings, conclusions, or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of DARPA.

References

[1] Svetlana Abramova, Artemij Voskobojnikov, Konstantin Beznosov, and Rainer Böhme. 2021. Bits under the mattress: Understanding different risk perceptions and security behaviors of crypto-asset users. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*.

[2] Bhupendra Acharya, Muhammad Saad, Antonio Emanuele Cinà, Lea Schönherr, Hoang Dai Nguyen, Adam Oest, Phani Vadrevu, and Thorsten Holz. 2024. Conning the crypto conman: End-to-end analysis of cryptocurrency-based technical support scams. In *IEEE Symposium on Security and Privacy*.

[3] Pieter Agten, Wouter Joosen, Frank Piessens, and Nick Nikiforakis. 2015. Seven months' worth of mistakes: A longitudinal study of typosquatting abuse. In *Proceedings of the Network and Distributed System Security Symposium*. Internet Society.

[4] Android. 2025. Access from internal storage. https://developer.android.com/training/data-storage/app-specific

[5] U.K. National Protective Security Authority. 2025. Social Engineering. https://www.npsa.gov.uk/security-campaigns/social-engineering-0

[6] Anirban Banerjee, Dhiman Barman, Michalis Faloutsos, and Laxmi N Bhuyan. 2008. Cyber-fraud is one typo away. In *IEEE INFOCOM*.

[7] Massimo Bartoletti, Salvatore Carta, Tiziana Cimoli, and Roberto Saia. 2020. Dissecting Ponzi schemes on Ethereum: identification, analysis, and impact. *Future Generation Computer Systems* (2020).

[8] Chainalysis. 2025. Anatomy of an Address Poisoning Scam. https://www.chainalysis.com/blog/address-poisoning-scam/

[9] Coinmetro. 2025. What Are Address Poisoning Attacks?: How to protect yourself from address poisoning attacks. https://www.coinmetro.com/learning-lab/address-poisoning-attacks

[10] European Union Council. 2025. Cybersecurity: social engineering. https://www.consilium.europa.eu/en/policies/cybersecurity-social-engineering/

[11] Mr. Creatonics. 2025. The Top 12 Best Ethereum Wallets (2025 Edition). https://coinsutra.com/best-etherum-wallets/

[12] Cube. 2025. Wallet security. https://www.cube.exchange/what-is/allowlistblocklist

[13] CyLab. 2025. Toxin Tagger (T-T): Real-time Detection System for Blockchain Address Poisoning. https://cryptotrade.cylab.cmu.edu/poisoning/ethereum

[14] Fred J Damerau. 1964. A technique for computer detection and correction of spelling errors. *Commun. ACM* 7, 3 (1964), 171–176.

[15] Alexis Deviney. 2025. The quest for the one true name across DNS and ENS. (2025).

[16] Farida Eleshin, Qi Sun, Mengzhe Ye, Sauvik Das, and Jason I Hong. 2025. Of Secrets and Seedphrases: Conceptual Misunderstandings and Security Challenges for Seed Phrase Management among Cryptocurrency Users. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*.

[17] ENS. 2025. ENS. https://ens.domains/

[18] ENS. 2025. How to Register a .eth name. https://support.ens.domains/en/articles/7882582-how-to-register-a-eth-name

[19] ENS. 2025. Name lifecycle. https://support.ens.domains/en/articles/8046877-name-lifecycle

[20] Ethereum.org. 2025. Introduction to smart contracts. https://ethereum.org/developers/docs/smart-contracts/

[21] Internet Crime Complaint Center (IC3) Federal Bureau of Investigation. 2025. 2024 Internet Crime Report. https://www.ic3.gov/AnnualReport/Reports/2024_IC3Report.pdf

[22] Michael Fröhlich, Maurizio Raphael Wagenhaus, Albrecht Schmidt, and Florian Alt. 2021. Don't stop me now! exploring challenges of first-time cryptocurrency users. In *Proceedings of the ACM designing interactive systems conference*.

[23] Google. 2025. Making the world's information safely accessible. https://safebrowsing.google.com/

[24] Shixuan Guan and Kai Li. 2024. Characterizing Ethereum address poisoning attack. In *ACM SIGSAC Conference on Computer and Communications Security*.

[25] Shixuan Guan and Kai Li. 2025. Ethereum Crypto Wallets under Address Poisoning: How Usable and Secure Are They? *arXiv preprint arXiv:2508.12107* (2025).

[26] Yongqi Guan, Yaman Yu, Tanusree Sharma, Kaihua Qin, Yang Wang, and Ye Wang. 2023. Examining User Perceptions of Stablecoins: Understandings and Risks. In *Posters at the Symposium on Usable Privacy and Security (SOUPS)*.

[27] Shuang Hao, Nick Feamster, and Ramakant Pandrangi. 2011. Monitoring the initial DNS behavior of malicious domains. In *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*. 269–278.

[28] hsts [n. d.]. HSTS Preload List Submission. https://hstspreload.org/

[29] Jianwei Huang, Sridatta Raghavendra Chintapalli, Mengxiao Wang, and Guofei Gu. 2025. Beyond Visual Confusion: Understanding How Inconsistencies in ENS Normalization Facilitate Homoglyph Attacks. In *Proceedings of the ACM on Web Conference*.

[30] Daiki Ito, Yuta Takata, Hiroshi Kumagai, and Masaki Kamizono. 2024. Investigations of top-level domain name collisions in blockchain naming services. In *Proceedings of the ACM Web Conference 2024*.

[31] Ooi Say Keat, Ooi Chai Aun, Yeap Jasmine AL, and Goh Tok Hao. 2021. Embracing Bitcoin: users' perceived security and trust. *Quality and Quantity* (2021).

[32] Vincent Kobel. 2025. Generating a usable Ethereum wallet and its corresponding keys. https://kobl.one/blog/create-full-ethereum-keypair-and-address/

[33] Katharina Krombholz, Aljosha Judmayer, Matthias Gusenbauer, and Edgar Weippl. 2016. The other side of the coin: User experiences with bitcoin security and privacy. In *International conference on financial cryptography and data security*. Springer.

[34] Tobias Lauinger, Abdelberi Chaabane, Ahmet Salih Buyukkayhan, Kaan Onarlioglu, and William Robertson. 2017. Game of registrars: An empirical analysis of {Post-Expiration} domain name takeovers. In *USENIX Security Symposium*.

[35] Tobias Lauinger, Kaan Onarlioglu, Abdelberi Chaabane, William Robertson, and Engin Kirda. 2016. WHOIS Lost in Translation: (Mis) Understanding Domain Name Expiration and Re-Registration. In *Proceedings of the Internet Measurement Conference*.

[36] Chaz Lever, Robert Walls, Yacin Nadji, David Dagon, Patrick McDaniel, and Manos Antonakakis. 2016. Domain-z: 28 registrations later measuring the exploitation of residual trust in domains. In *IEEE symposium on security and privacy (SP)*.

[37] Kai Li, Darren Lee, and Shixuan Guan. 2023. Understanding the cryptocurrency free giveaway scam disseminated on twitter lists. In *IEEE International Conference on Blockchain*.

[38] Xigao Li, Anurag Yepuri, and Nick Nikiforakis. 2023. Double and nothing: Understanding and detecting cryptocurrency giveaway scams. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*.

[39] Enze Liu, George Kappos, Eric Mugnier, Luca Invernizzi, Stefan Savage, David Tao, Kurt Thomas, Geoffrey M Voelker, and Sarah Meiklejohn. 2024. Give and take: An end-to-end investigation of giveaway scam conversion rates. In *Proceedings of the ACM on Internet Measurement Conference*.

[40] MetaMask. 2025. MetaMask. https://metamask.io/

[41] MetaMask. 2025. MetaMask and ChainPatrol protect users with phishing warnings. https://metamask.io/news/metamask-chainpatrol-protect-users-with-phishing-warnings

[42] MetaMask. 2025. MetaMask Security Alerts by Blockaid: the new normal for a safer transaction experience. https://metamask.io/news/metamask-security-alerts-by-blockaid-the-new-normal-for-a-safer-transaction

[43] MetaMask. 2025. Product Spotlight: Privacy Preserving Features in MetaMask. https://metamask.io/news/product-spotlight-privacy-preserving-features-in-metamask

[44] Najmeh Miramirkhani, Timothy Barron, Michael Ferdman, and Nick Nikiforakis. 2018. Panning for gold.com: Understanding the dynamics of domain dropcatching. In *Proceedings of the Web Conference*.

[45] Tyler Moore and Benjamin Edelman. 2010. Measuring the perpetrators and funders of typosquatting. In *International Conference on Financial Cryptography and Data Security*. Springer.

[46] Muhammad Muzammil, Zafir Ansari, Nick Nikiforakis, and Darin Johnson. 2026. Echoes of the Past: Detecting and Classifying Re-registered Domains in Enterprise DNS Traffic. In *Workshop on Measurements, Attacks, and Defenses for the Web (MADWeb)*.

[47] Muhammad Muzammil, Abisheka Pitumpe, Xigao Li, Amir Rahmati, and Nick Nikiforakis. 2025. The poorest man in babylon: A longitudinal study of cryptocurrency investment scams. In *Proceedings of the ACM on Web Conference*.

[48] Muhammad Muzammil, Harsh Shah, and Nick Nikiforakis. 2026. No Wallet for Old Tricks. https://pragseclab.github.io/no-wallet-for-old-tricks/

[49] Muhammad Muzammil, Zhengyu Wu, Aruna Balasubramanian, and Nick Nikiforakis. 2024. Panning for gold. eth: Understanding and Analyzing ENS Domain Dropcatching. In *Proceedings of the ACM on Internet Measurement Conference*.

[50] Muhammad Muzammil, Zhengyu Wu, Lalith Harisha, Brian Kondracki, and Nick Nikiforakis. 2024. Typosquatting 3.0: Characterizing squatting in blockchain naming systems. In *APWG Symposium on Electronic Crime Research (eCrime)*. IEEE.

[51] Palo Alto Networks. 2025. What Are Malicious Newly Registered Domains? https://docs.netskope.com/en/rbi-category-definitions

[52] Palo Alto Networks. 2025. What Are Malicious Newly Registered Domains? https://www.paloaltonetworks.com/cyberpedia/what-are-malicious-newly-registered-domains

[53] U.S. Department of State. 2025. Understanding the Dangers of Social Engineering. https://www.state.gov/understanding-the-dangers-of-social-engineering

[54] Pujan Paudel and Gianluca Stringhini. [n. d.]. LOKI: Proactively Discovering Online Scam Websites by Mining Toxic Search Queries. *Newtwork and Distributed System Security Symposium (NDSS)* ([n. d.]).

[55] Protos. 2025. Refund of $70M address poisoning scam ongoing, over 50% returned. https://protos.com/refund-of-70m-address-poisoning-scam-ongoing-over-50-returned/

[56] Michael Sikorski. 2025. Social Engineering on the Rise — New Unit 42 Report. https://www.paloaltonetworks.com/blog/2025/07/social-engineering-rise-new-unit-42-report/

[57] Gilberto Atondo Siu and Alice Hutchings. 2023. "Get a higher return on your savings!": Comparing adverts for cryptocurrency investment scams across platforms.

In *IEEE European symposium on security and privacy workshops (EuroS&PW)*. IEEE.
[58] Gilberto Atondo Siu, Alice Hutchings, Marie Vasek, and Tyler Moore. 2022. "Invest in crypto!": An analysis of investment scam advertisements found in Bitcointalk. In *APWG symposium on electronic crime research (eCrime)*. IEEE.
[59] Johnny So, Najmeh Miramirkhani, Michael Ferdman, and Nick Nikiforakis. 2022. Domains do change their spots: Quantifying potential abuse of residual trust. In *IEEE Symposium on Security and Privacy (SP)*.
[60] Johnny So, Iskander Sanchez-Rola, and Nick Nikiforakis. 2025. Lost in the Mists of Time: Expirations in DNS Footprints of Mobile Apps. (2025).
[61] Jeffrey Spaulding, DaeHun Nyang, and Aziz Mohaisen. 2017. Understanding the effectiveness of typosquatting techniques. In *Proceedings of the ACM/IEEE Workshop on Hot Topics in Web Systems and Technologies*.
[62] Chris Stokel-Walker. 2025. Scammers are making thousands of dollars through blockchain typosquatting. https://www.fastcompany.com/91228285/scammers-are-making-thousands-of-dollars-through-blockchain-typosquatting
[63] Janos Szurdi, Balazs Kocso, Gabor Cseh, Jonathan Spring, Mark Felegyhazi, and Chris Kanich. 2014. The long {"Taile"} of typosquatting domain names. In *USENIX Security Symposium*.
[64] Taro Tsuchiya, Jin-Dong Dong, Kyle Soska, and Nicolas Christin. 2025. Blockchain Address Poisoning. In *USENIX Security Symposium*.
[65] Yi-Min Wang, Doug Beck, Jeffrey Wang, Chad Verbowski, and Brad Daniels. 2006. Strider Typo-Patrol: Discovery and Analysis of Systematic Typo-Squatting. *SRUTI* (2006).
[66] Kristin Weber, Andreas E Schütz, Tobias Fertig, and Nicholas H Müller. 2020. Exploiting the human factor: Social engineering attacks on cryptocurrency users. In *International Conference on Human-Computer Interaction*. Springer.
[67] Pengcheng Xia, Haoyu Wang, Zhou Yu, Xinyu Liu, Xiapu Luo, Guoai Xu, and Gareth Tyson. 2022. Challenges in decentralized name management: the case of ENS. In *ACM Internet Measurement Conference*.
[68] Guoyi Ye, Geng Hong, Yuan Zhang, and Min Yang. 2024. Interface Illusions: Uncovering the Rise of Visual Scams in Cryptocurrency Wallets. In *Proceedings of the ACM Web Conference*.

A User Interface Screens

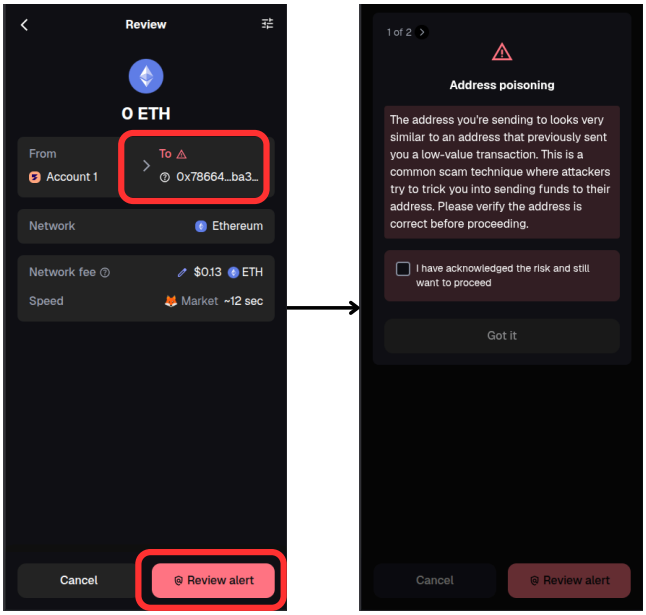


Figure 4: Overview of the address poisoning detection workflow integrated into MetaMask. The wallet evaluates the entered recipient address against the user's transaction history and displays a warning when a suspicious match is detected.

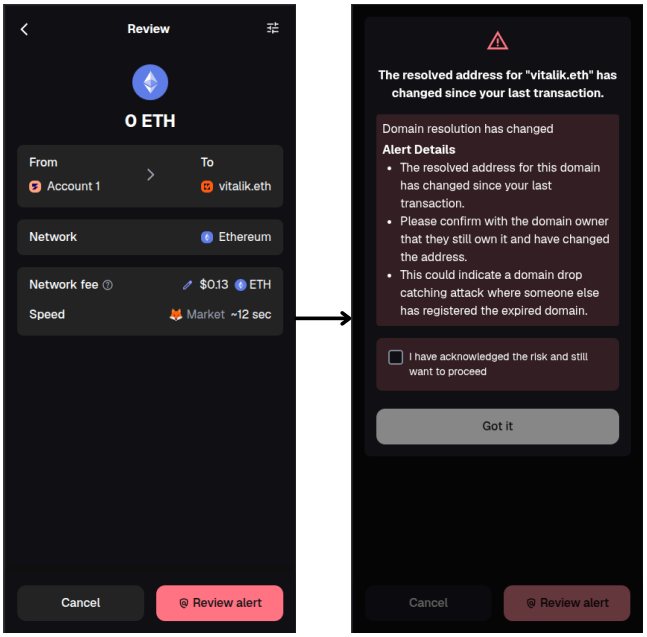


Figure 6: Screenshots illustrating the integrated MetaMask's ENS drop-catching detection mechanism. When a user enters an ENS domain, the wallet checks whether the domain has expired and been re-registered, and surfaces a warning if the domain appears to have changed ownership.

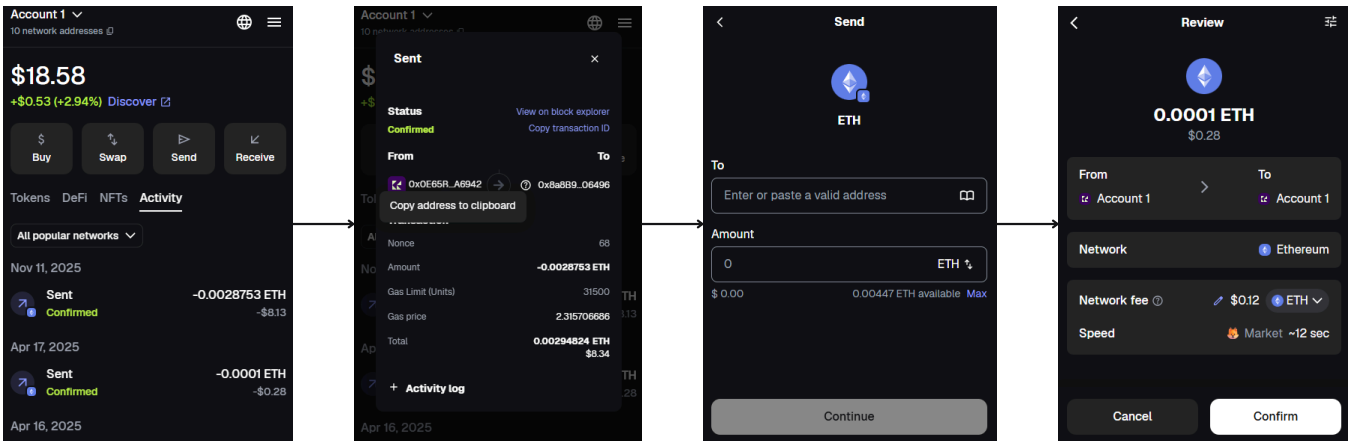


Figure 5: MetaMask screens as a user is performing a transaction, upto the transaction confirmation page without any presence or detection of malicious activity

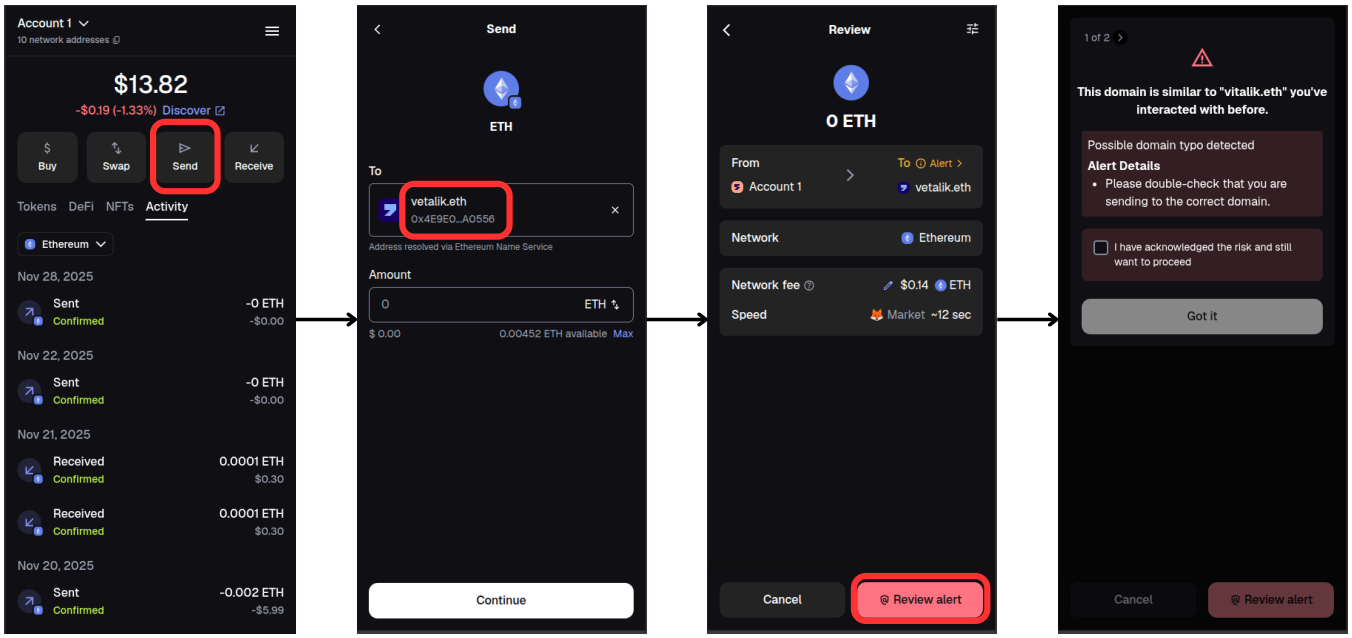


Figure 7: Screenshots illustrating the integrated MetaMask's ENS typosquatting detection mechanism. When a user enters an ENS domain, the wallet checks for similarity against previously used domains and surfaces a warning if a suspicious match is detected.