Understanding End-User Perception of Transfer Risks in Smart Contracts

Yustynn Panicker
yustynn@gmail.com
Singapore University of Technology and Design
Singapore

Ezekiel Soremekun
ezekiel_soremekun@sutd.edu.sg
Singapore University of Technology and Design
Singapore

Sudipta Chattopadhyay
sudipta_chattopadhyay@sutd.edu.sg
Singapore University of Technology and Design
Singapore

Sumei Sun
sunsm@i2r.a-star.edu.sg
Institute for Infocomm Research ASTAR
Singapore

ABSTRACT

Blockchain smart contracts are increasingly used in critical use cases (e.g., financial transactions). Thus, it is pertinent to ensure that their end-users understand risks in attempting token transfers. Addressing this, we investigate end-user comprehension of five transfer risks (e.g. the end-user being blacklisted) in the most popular Ethereum contract, USD Tether (USDT), and their prevalence in other top ERC-20 contracts. First, we conducted a user study investigating end-user comprehension of transfer risks in USDT with 110 participants. Second, we performed source code analysis of the next top (78) ERC-20 smart contracts to identify the prevalence of these risks. Study results show that the majority of end-users do not comprehend some real risks, and confuse real and fictitious risks. This holds regardless of participants' self-rated programming and Web3 proficiency. Source code analysis demonstrates that examined risks are prevalent in up to 19.2% of the top ERC-20 contracts.

CCS CONCEPTS

• Human-centered computing → User studies; **• Security and privacy → Human and societal aspects of security and privacy**; *Social aspects of security and privacy.*

KEYWORDS

smart contract, transfer risk, user perception, ethereum, erc-20

ACM Reference Format:
Yustynn Panicker, Ezekiel Soremekun, Sudipta Chattopadhyay, and Sumei Sun. 2025. Understanding End-User Perception of Transfer Risks in Smart Contracts. In *CHI Conference on Human Factors in Computing Systems (CHI '25), April 26-May 1, 2025, Yokohama, Japan.* ACM, New York, NY, USA, 21 pages. https://doi.org/10.1145/3706598.3713887

CHI '25, April 26-May 1, 2025, Yokohama, Japan

ACM ISBN 979-8-4007-1394-1/25/04...$15.00
https://doi.org/10.1145/3706598.3713887

1 INTRODUCTION

Blockchain smart contracts have become increasingly important in our society. Common applications of smart contracts include financial contracts, and gaming [80]. Notably, the top ten public blockchains implement smart contracts [6], including Ethereum [28], Polygon [4] and Binance Smart Chain [7]. Considering the critical use cases of smart contracts (e.g., in financial transactions), *it is important to understand the end-users' comprehension of the transfer risks in smart contracts.* Such risks (e.g., transaction risks) may have severe consequences, e.g., financial loss and unexpected outcomes.

To the best of our knowledge, this paper presents the first comprehensive study to investigate end-users' understanding of transfer risks of smart contracts. Broadly, our research methodology comprises of two key components, namely *(i)* a *user study* examining smart contract transfer risks with 110 end-users using the Etheruem-based USDT contract, and *(ii) source code analysis* of (78) Etheruem-based ERC-20 contracts investigating the generalization of the examined transfer risks in *(i)*. Figure 1 illustrates our research methodology.

In this work, we investigate Ethereum-based contracts since Ethereum has the largest total value locked (TVL) [17] by far (USD 25B [6]) among the top ten public blockchains. Our source code analysis (in Figure 1) focuses on smart contracts that implement the ERC-20 standard [1] for fungible tokens. ERC-20 was chosen as contracts implementing the ERC-20 standard received the largest transaction volume during our analysis. Concretely, among the top 500 recipient addresses, USDT together with the other 78 ERC-20 contract addresses studied in this work, accounted for 20.0% of transaction volume. We design our *user study* (in Figure 1) using the USDT contract since it is the most popular smart contract, accounting for 12.7% of transaction volume to the top 500 recipient addresses in a three-month period.

Our inspection of the USDT contract revealed various potentially surprising features beyond the ERC-20 specification [1]. These include the ability to *blacklist users*, *pause the contract*, *upgrade the contract arbitrarily* and *set fees on user transfers of USDT*. Figure 2 illustrates some of these risks, capturing the sequence of events and the user interface (UI) flow (using MetaMask UI) for a *failed transaction* due to the contract being *paused*, the user *blacklisted* or arbitrary contract *upgrade.* We focus on these risks because they affect the transfer outcomes, thus directly impacting the financial objective of the smart contract end-user. To check if end-users are

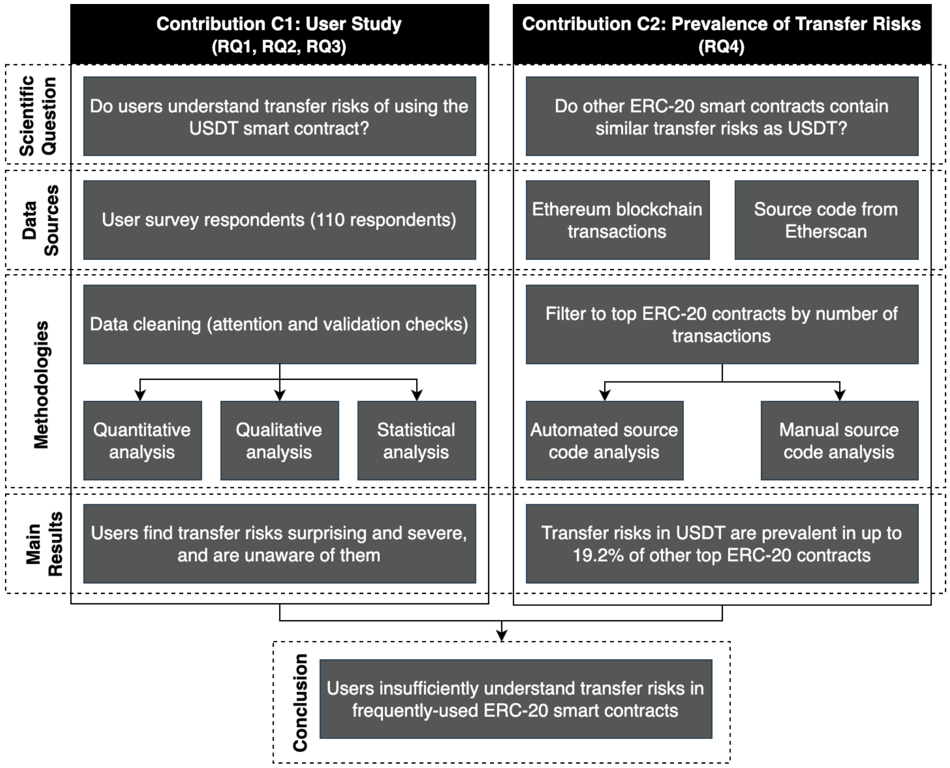


Figure 1: Overview of Research Methodology

aware of these risks, we conducted a user study with 110 smart contract users ("User Study" in Figure 1). Through a series of quantitative and qualitative analysis, we discover that most users are *unaware* of and *surprised* by these risks. More importantly, end-users perceived the studied risks (e.g., the three risks illustrated in Figure 2) to be *severe*. Our statistical analysis further shows that our findings hold for different user groups (e.g., self-rated programmers vs. non-programmers.)

To extend our analysis beyond the USDT contract, ("Generalizability Analysis" in Figure 1), we show that the transfer risks examined in the USDT contract extend to other popular ERC-20 smart contracts. To this end, we performed both manual and automated analysis of the next 78 most frequently used ERC-20 contracts (excluding USDT, since we performed a manual analysis of USDT for the user study). Moreover, manual analysis of these 78 ERC-20 contracts revealed additional potential end-user risks beyond the ones considered in our user study. Overall, our findings concretely point to the insufficiency of user understanding in dealing with a variety of smart contract risks across the most popular contracts.

Our study is unique in that it deals with the end-users' perception directly. Several existing works have focused on programming languages [37, 71] and tools [33, 59, 79, 92] for smart contract *developers*, but these works do not target *end-users*. Concurrently, existing user studies with smart contract aim to either understand specific user preferences and misconceptions (e.g., cryptographic key) [27, 60] or to validate certain technologies (e.g., user notice in the code [51]). Our study is orthogonal to these approaches, as instead of focusing on a specific technology and user preference, we aim to broadly investigate the users' comprehension through a widely used smart contract and interface (USDT and MetaMask).

Concretely, we make the following contributions:

- **(C1) User Study:** To the best of our knowledge, we present the first user study to investigate the end-users' understanding of smart contract transfer risks, through USDT (section 4). We provide detailed analysis of the study responses, including statistical analysis where relevant. Notably, our analysis reveals that *up to 71.8% of users believe that contract upgrade and blacklisting are the most severe and most surprising transfer risks.* Moreover, only up to 35.8% of users found the MetaMask UI sufficient to understand transfer risks (see Figure 2). In comparison, more than twice as many (82.7% of) end-users understand the successful transfer outcome (section 5). Statistical analysis reveals that neither self-rated programming nor Web3 proficiency significantly influence end-users' ratings of transfer risks and their comprehension of MetaMask UI flows for successful and failing transfer outcomes (section 5).
- **(C2) Prevalence of transfer risks:** We investigate the prevalence of transfer risks in other ERC-20 contracts through automated and manual source code analysis of the top 78 contracts (excluding USDT) (section 4). This revealed that examined transfer risks are 19.2% prevalent. Additionally, our manual analysis revealed three additional transfer risks (section 5).

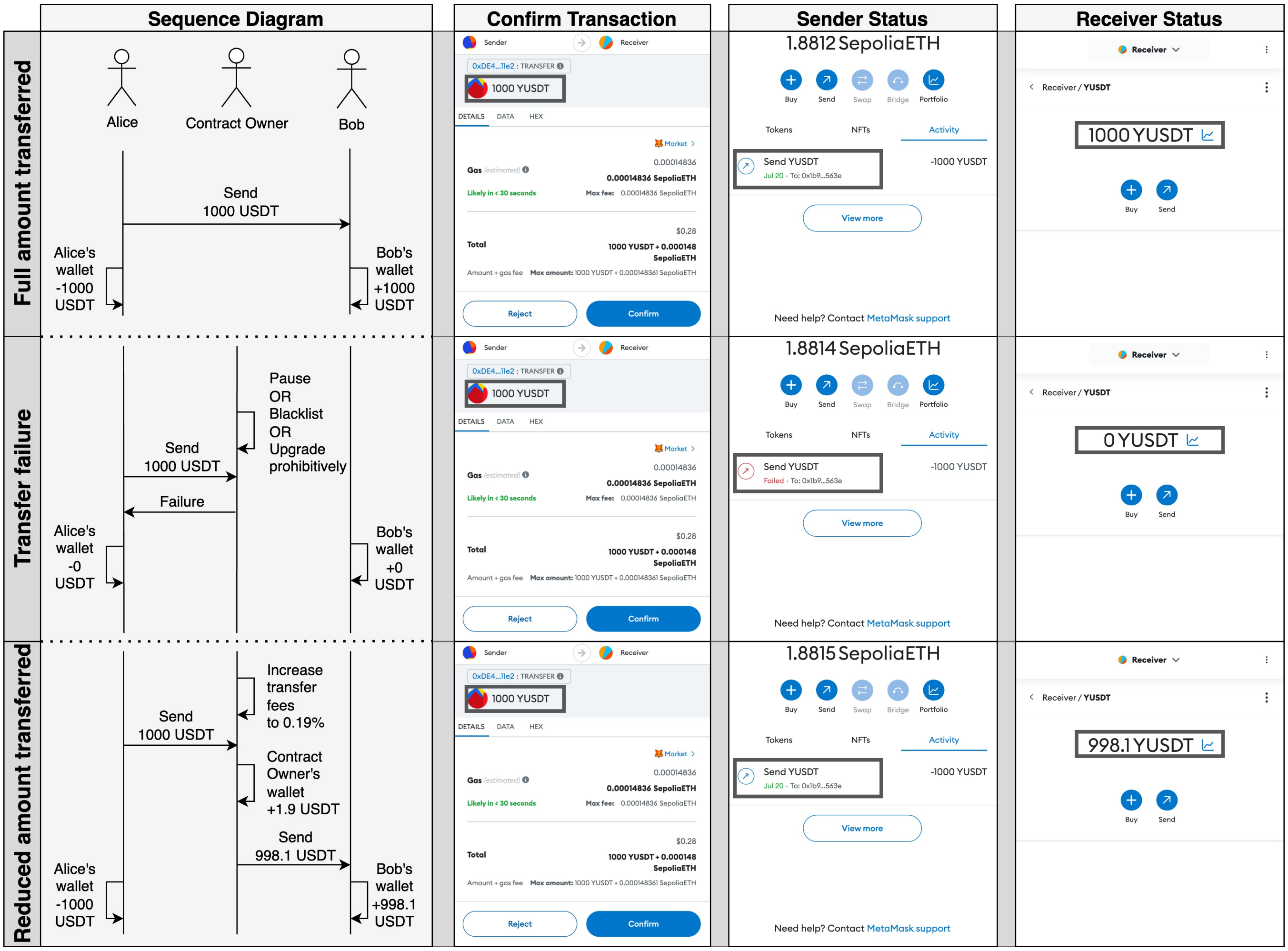


Figure 2: Sequence of events and corresponding MetaMask flow for a user's failure to make a USDT transfer. The MetaMask flow shows our functionally-equivalent clone of the USDT contract named YUSDT, with bounding boxes highlighting relevant aspects. All scenarios are based on the USDT source code, reflecting intentional features and not fraudulent behavior.

We discuss key takeaway points and future outlook based on the study results (section 6) before highlighting possible threats in our study (section 7) and concluding (section 8).

2 BACKGROUND AND TERMINOLOGY

Ethereum and Smart Contracts: A blockchain is a mechanism to arrive at an agreement among a set of decentralised actors about the order of events recorded in an append-only data structure [58]. *Ethereum* [28] is a public blockchain with over 238M unique addresses [10] and a market capitalization of over USD 230B [9]. In contrast to Bitcoin, Ethereum is able to execute Turing-complete code known as smart contracts, a term coined prior to blockchain by Szabo [77]. Ethereum is known as the start of the second generation of blockchains, which often have smart contract capabilities.

Tokens: One major use case of Ethereum smart contracts is implementating auxiliary *virtual tokens* [1–3]. These tokens have a variety of uses e.g., for implementing currencies, for marking asset ownership (such as with non-fungible tokens - NFTs), or for conferring voting power in some community.

ERC-20: The programmatic interfaces for tokens employ agreed-upon standards in the Ethereum community. ERC-20 is such an interface standard [1] for fungible tokens. During our analysis period, the ERC-20 standard [1] for fungible tokens was the most used, accounting for 20.0% of all transactions involving the top 500 recipient addresses.

Transfer Risks: By transfer risks, we refer to unexpected events which may not match users' expectations when transferring tokens from one Ethereum address to another.

USDT: The dominant share of ERC-20 transactions went to the USDT contract (12.7% of transactions to the top 500 recipient addresses), which manages the USDT stablecoin. A stablecoin is a store of value in which its price relative to some fiat currency (i.e., the US dollar) is meant to stay stable. USDT is an asset-backed stablecoin which has a market capitalization of over USD 83B [16].

Light Wallets and MetaMask: End-users typically use software known as *light wallets* to interact with blockchains. For Ethereum, a popular interface is the MetaMask wallet [8]. The advantage of such wallets is their ease-of-use (the GUI removes the need for user code) and ease-of-setup (no need to download the entire blockchain). The standardized interface of tokens allows end-user wallets to programmatically extract salient features of the contract (e.g., the token symbol, the function enabling fund transfer and the ability to retrieve a user's balance) for presentation to the end-user.

Software Engineering Relevance: Our study is directly related to the end-users' perception of risks associated with software systems, in particular, smart contracts. Since smart contracts are relatively new and complex, there is a *lack of empirical studies or automated methods to enable the understanding of end-users' perception of transfer risks.* To this end, our work contributes to the broader field of *(empirical) software engineering for smart contracts.* Moreover, our study *not only* highlights the concrete risks, but also provides a groundwork for automated detection of transfer risks in the source code of smart contracts. This also allows for the development of better wallet interfaces to highlight the transfer risks to end users. Besides, our study informs the need for end-user-oriented software pipeline for smart contracts. For instance, it motivates the development of (a) techniques (e.g., code/UI annotations) to explain the associated transfer risks to end users and (b) automated testing and mitigation methods to discover, avoid and prevent such transfer risks.

3 RELATED WORK

Table 1 illustrates the overall positioning of our work with respect to existing literature. In the following, we further categorize the current works based on their scope and objective.

User Perspectives on Blockchain Technology (C1): Current works explore a variety of user perspectives, encompassing first-time users, experienced users, miners and people who do not use blockchain. The majority of these studies focus either on the perceptions and behaviors relating to trust, security and privacy [24, 42, 57, 65, 70, 74, 82], or on wallet and key management perceptions [44, 61, 83, 93]. A minority of user studies engage in more focused topics. These include creator and holder perceptions of NFTs [88], user perceptions of third-party security audits [41, 53] and user perception of the category of sandwich attacks [86]. Most related to our topic, two recent studies [49, 50] concern user perceptions of stablecoins, but they study stablecoins as a broad category rather than focusing on the implemented transfer risks of a particular stablecoin. Many of these works highlight incorrect user understanding, either by showcasing inaccuracies in user mental models [60, 61, 69] or highlighting misunderstandings which underpin some user statements [50, 83]. While there has been work which encompasses perceptions on the breadth of Web3, including the smart contract layer, to the best of our knowledge, there has been no work which studies user understanding of a particular, widely used smart contract. Our work thus distinguishes itself from the existing body of user studies in two ways: First, by focusing on a specific yet widely used smart contract (USDT), and second, by examining the category of transfer risks. Additionally, we extend our findings through blockchain data analysis. This is methodologically uncommon across existing studies, which are often centred on user perspectives alone.

Human Factors in Smart Contracts (C1, C2): Our findings are in line with prior works that emphasized the user aspects in blockchain design and improving the design of wallets accordingly [45, 64]. However, our goal is orthogonal to approaches that involved users in the validation of smart contracts, e.g., via user experience design [55], generation of warnings [46], programming language design [37] and user notice generation [51]. Specifically, unlike previous works, which aim to validate a new technology, our work (*see* section 6) identifies the risks and gaps in smart contract usage using a widely used smart contract (USDT/MetaMask). Our work also complements approaches that target users' specific behaviors e.g., key management preferences [27], misconceptions of cryptographic keys and blockchain encryption [60].

Programming Languages and Tools for Smart Contracts (C2): In the last few years, researchers have emphasized the need for fixing defects in smart contract code [33]. To this end, prior works explore security practices of smart contract developers [84], new programming language constructs for safe smart contract [37, 38, 71], comprehension of smart contract code [32, 72, 73, 87], detection of API documentation errors [96], verification of smart contract fairness [59] and verification of smart contracts written in a declarative language [31]. Other researchers have also conducted oracle deviation analysis [40], testing based on symbolic execution [63], code clone detection via code embedding [48], verification with the aid of specification tailored to smart contract [26] and static analysis based bug hunting [75, 79, 91]. Empirical studies have also analyzed the effectiveness of existing tools [29, 90]. Our work aligns with these works under the broader category of smart contract risks. However, it is distinguished in that the transfer risks we study are intended behaviors, and therefore unrelated to the implementation bugs that the majority of works in this category study.

Security, Privacy and Fairness of Smart Contracts (C1, C2): To avoid exploitation of security and privacy concerns in smart contract [36, 66], several works have been proposed including threat modeling [25], recommendation and validation of secure programming pattern [68], domain-specific and privacy-preserving services [85], security incident response [67], static analysis for vulnerability detection [35], [76], dynamic-analysis-based online defense [78], investigating patterns of library misuse [52], simulating user behavior with bots to unveil defects arising from multi-user interactions [81], smart contract code repair [92, 94], and fair, transparent use of smart contracts [39]. Our study is orthogonal to these works, as we investigate *smart contract transfer risks for end-users* instead of design or coding flaws.

Transfer Risks (C1, C2): There is substantial research work in detecting inconsistent and unexpected behaviors in ERC-20 tokens. TokenScope [34] automatically detects inconsistencies by comparing the ERC-20 specification, events fired during usage and their effects on relevant data structures. Another work [54] proposes a classifier for identifying *administration patterns* in existing tokens. There is an overlap in the transfer risks examined in our work and existing work (e.g., Arbitrary Mint and Destroy User Funds), but our work includes unexamined transfer risks (User Blacklist and Insufficient Funds). Besides, contrary to prior works on detecting and avoiding transfer risks, we examine the end-users' perception of such risks.

4 RESEARCH METHODOLOGY

In this work, we pose the following research questions:

RQ1 Risk Perception: How do end-users perceive the transaction risks of USDT smart contracts?

RQ2 Understanding of User Interface: Does the MetaMask wallet effectively inform users of possible transfer outcomes?

RQ3 Smart Contract Understanding: How do smart contract end-users educate themselves? How do they perceive their own comprehension of how smart contracts work?

RQ4 Generalizability of Risks: Do the transfer risks of USDT contract occur in other ERC-20 contracts?

The first three research questions lead to contribution **C1**, while **RQ4** leads to contribution **C2**. **RQ1** describes the central question of transfer risk perception in the most used Ethereum smart contract (USDT). Building on this, **RQ2** investigates how the MetaMask UI may lead to incorrect transfer risk perception. In a similar vein, **RQ3** aims to deepen our understanding of factors that mediate transfer risk perception by investigating how users comprehend smart contracts, as well as their confidence in their comprehension. Finally, **RQ4** studies the generalization of these transfer risks to other ERC-20 contracts, extending our findings beyond USDT.

4.1 User Study Design (RQ1, RQ2, RQ3)

Survey Questionnaire: The user study questionnaire contains 196 questions implemented on Google Forms [12]. It begins with a consent form explaining study goals and obtaining participants' consent to *anonymously* collect response data. In the first part of the survey, we posed questions about the demography of respondents (e.g., age and profession) and their knowledge/expertise level. For instance, we asked about the participants' knowledge of smart contracts, USDT, stablecoins and programming. The second part of the survey gathers information about USDT user behavior and risk awareness. It contains questions about the users perception of transaction outcomes and transfer risks, the usabilty of the MetaMask user interface, and user preferences for alternative descriptions of smart contract behavior. We provide a copy of the survey questionnaire for scrutiny and reuse (see section 8).

Recruitment and Compensation: The study was conducted using Prolific [14], a well-known platform for conducting industrial and research surveys. We chose this platform because it allows to pre-screen for participants with experience or expertise in specific domains, in our case smart contracts. Specifically, we screened for participants that are fluent in English language, and completed at least secondary school education. We also require that participants have high quality answers (approval rating between 98 to 100, with at least 100 prior submissions on the platform) in prior surveys and have a knowledge of cryptocurrency or cryptocurrency exchanges. All participants gave consent prior to participation, and were provided with multiple avenues for feedback during and after the survey. After the study, we sent a draft of the paper to all participants explicitly informing them about true and fictitious transfer risks. Overall, we recruited 110 respondents for the final study and paid each participant £7.04. The main study took 44 days to complete starting from 30th January 2023.

Pilot Study: We conducted the first pilot study for three days (from 11th to 13th January 2023) with four researchers and librarians in order to obtain feeedback on the design and identify unclear questions in the study questionnaire. The feedback enabled us to add several free-text questions to elicit reasons for users' responses and to allow participants to provide additional information about risk comprehension. Using the revised questionnaire, we conducted a second pilot study for a day with 10 participants from Prolific (paid £7.04) who had used smart contracts before. Analyzing these responses informed the inclusion of new questions, e.g., questions on comprehension of the MetaMask user interface (UI) flow.

Demographics: The survey was conducted on 110 respondents. Participants' age is from 18 to 64 years, most (54.6%) being 25-34 years old and least (4.6%) being between 55 to 64 years old. Participants are from over 21 different sectors, the top three being Computing or IT (23.6%), Engineering or Manufacturing (11.8%), and Students (8.2%). Geographically, the respondents are from four continents and 18 countries. The top three countries were the United States (26.4%), South Africa (22.7%) and the United Kingdom (17.3%).

Validating Smart Contract Usage: The main threat to construct validity is that respondents may falsely claim to be smart contract users, e.g., because they transact on intermediary cryptocurrency exchanges. We mitigate this by (a) pre-screening for users (on Prolific) who were knowledgeable about cryptocurrency, (b) asking validation questions about smart contract usage, (c) notifying respondents that transferring from an exchange to their wallet is not smart contract usage, and (d) asking respondents to rate themselves on their Web3 proficiency. We note that most (94%) of the respondents have at least some Web3 proficiency and passed our validation questions.

Proficiency Validation: The self-rated Web3 and programming proficiencies of participants were verified with validation questions based on verifying the participants' knowledge of common concepts (e.g., *Please give an example of a low-level programming language*). However, we note that these validation questions do not provide full confidence in self-rated scores which are high, particularly for programming – a more skill-based proficiency.

Quantitative Response Data Analysis: We report the number of respondents that chose each option (e.g., severity level score on Likert scale) as well as the percentage of respondents that chose the option (*see* Table 3, Table 4 and Table 5). We further categorised each likert scale response into three levels, e.g., Aware (score 1-2), Somewhat Aware (3) and Unaware (4-5). The total percentages may be below 100% (*in* Table 2 and Table 6) since some responses were discarded due to clear misunderstandings of the question by the participant, confusing reasons or incorrect assessments of the presented scenario. Such responses were not categorized, but are mentioned when worth highlighting (section 6, Table 2 and Table 6).

Qualitative Coding Protocol: To analyze free-text questions for qualitative results, we use a coding protocol [30] involving at least two researchers. We extracted qualitative results for four (4) real transaction risks (**RQ1**) and all MetaMask flow evaluations (**RQ2**). Our coding protocol involved one researcher manually deriving the initial response categorizations, which was then validated by

Table 1: Comparison of our work vs most relevant related works. ●: Full consideration, ◐: Partial consideration, ○: Exclusion. The comparison categories are derived based on our contributions (C1/C2): concerns smart contract transfer risks ("Transfer Risks"), concerns user perception ("User Perception"), concerns smart contract implementation bugs/vulnerabilities ("Bugs or Vulnerabilities"), utilizes user study ("User Study"), analyzes blockchain data or smart contract source code ("Blockchain Data or Code").

Works	Transfer Risks (C1, C2)	User Perception (C1)	Bugs or Vulnerabilities	User Study (C1)	Blockchain Data or Code (C2)
This Work	●	●	○	●	●
Froehlich et al. [42]	◐	●	●	●	○
Si et al. [74]	◐	●	◐	●	○
Chen et al. [34]	◐	●	◐	○	●
[49, 88]	◐	●	○	●	○
Chaliasos et al. [29]	◐	○	●	○	●
Ivanov et al. [54]	◐	○	○	○	●
Yin et al. [90]	○	●	●	●	●
Chen et al. [32]	○	●	◐	●	●
Huang et al. [53]	○	●	◐	●	○
[51, 72, 73, 86]	○	●	○	●	●
[24, 25, 43, 44, 47, 50, 60, 61, 65, 69, 70, 82, 93]	○	●	○	●	○
Voskobojnikov et al. [83]	○	●	○	○	○
[52, 81, 95]	○	○	●	○	●
Wen et al. [87]	○	○	◐	○	●
Daian et al. [39]	○	○	○	○	●

another researcher and conflicts were resolved in the coding to agree on the categorization of responses. This process took approximately 30 hours in total. In general, we categorized reasons with associated scores above three (3) as positive, and below three (3) as negative. The presented results (in section 5, e.g., Table 2 and Table 6) are the consensus after coding and validation.

Examined Real Risks: We examined all five transfer risks in the USDT contract to investigate users' perceptions of real risks. The first four risks stop users from transferring USDT.
(1) Contract Pause: The contract being *paused*.
(2) User Blacklist: The user being *blacklisted*.
(3) Contract Upgrade: The contract being *upgraded* to a new, arbitrary contract, contrary to the original implementation.
(4) Insufficient Funds: *insufficient funds* in the user's account.
(5) Transfer Fee Increase: Increased fee parameters (currently zero) reducing the amount the receiver obtains from the sender.

Except for the insufficient funds risk, these identified risks represent ways in which a USDT transfer could fail beyond the ERC-20 specification.

Fake/Fictitious Risks: To further establish users' misunderstandings of smart contracts, we constructed the following five fake risks based on misconceptions that we suspected users may hold about the design and implementation of Ethereum and USDT:
(1) Consortium Reject: A user's inability to transfer USDT due to a consortium of Tether *users voting to reject* the transfer. This risk is based on the possibility that users may have incorrect concepts of decentralization, potentially due to misunderstandings [60] or centralization-decentralization tradeoffs [88], which may lead them to believe that majority vote for acceptance of token transfers is plausible. This feature is not in any ERC-20 source code we analyzed.
(2) Government Block: The transaction fails due to a *government blocking* it. The intuition for this risk is that users might incorrectly anchor expectations on the centralized financial systems [60] which they may be more familiar with or assume cooperation with regulatory bodies [50] that goes beyond what is possible on the smart contract layer. This is not possible as Ethereum's consensus algorithm [28] was designed to prevent centralized control.
(3) Receiver Reject: The *receiver rejects* the transfer. The risk is designed to check if users might believe they have a level of control over their wallets which they do not possess, possibly due to misunderstandings about permissions and decentralization [60]. This ability was not present in any ERC-20 source code we analyzed.
(4) Partial Funds: Sending a larger amount (e.g., 10 USDT) than a sender owns (e.g., 5 USDT) results in the receiver receiving only the sender's wallet amount (i.e., 5 USDT, not 10 USDT). The intuition behind this risk is a check for whether, perhaps due to lack of knowledge, users might have incorrect assumptions about the implementation of the transfer function. While such an implementation contradicts the ERC20 specification [1], prior work [34] has noted many instances in which this specificaion is not well-implemented. In reality, the Tether implementation would correctly cause the transaction to simply fail.
(5) Gas Fee Increase: Receivers receive less USDT due to *fluctuations in gas fees*. This risk is in line with numerous prior works [60, 69, 83] which demonstrate that users often find gas fees difficult to understand. This is misconceived since gas fees, paid in ETH (not USDT), cannot reduce the USDT transferred.

In order to avoid biasing participants during the survey, participants were only informed that fake risks were included after the survey. They were then informed about which risks were real and which were fictitious.

Collecting User Interface (UI) Flows: In our user study, we used the MetaMask light wallet to present users with screenshots of a standard transfer flow for USDT. To achieve this, we cloned the USDT smart contract and deployed it on the Ethereum Sepolia test network [18] under the token name YUSDT. Our ownership over the deployed contract allowed us to tweak YUSDT parameters

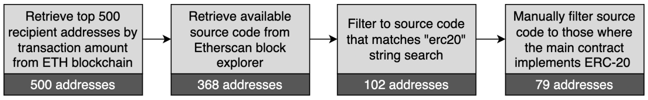


Figure 3: Source code retrieval process

in the same manner possible in the actual USDT contract by its owner. This, in turn, enabled observation of the MetaMask interface differences under varied contract parameters. To avoid bias, we discard responses that may be due to incomplete UI flows, e.g., missing final screenshot or intermittent pop-ups.

Evaluating Transaction Outcomes: We examined three main transaction outcomes from the MetaMask UI flow:
(1) Full amount transferred – the full transfer amount reaching the recipient.
(2) Reduced amount transferred – a reduced transfer amount reaching the recipient.
(3) Failure of the transfer – no change occurring in both the sender and recipient's wallet.
For each outcome, we asked the following questions:
(1) Was it (the UI) sufficient to **discover** the possibility of the outcome (with one - *Not sufficient at all (I am completely uninformed about the possible outcomes)*, five - *Fully sufficient (I fully understand the possible outcomes)*)
(2) Was it (the UI) sufficient to **understand** reasons behind those outcomes arising (one - *Not sufficient at all (I am completely uninformed about the possible outcomes)*, five - *Fully sufficient (I fully understand the possible outcomes)*)

Each question was accompanied with a text field asking the respondent to describe the reason for their choice.

Metrics and Measures: For each examined (real or fake) risk, we asked participants to evaluate their *awareness*, *surprise* and perceived *severity* of the risk on Likert scales as follows:
(a) Unawareness [1]*:* score "one (1)" as *I fully knew this could happen* to "five (5)" as *I had no idea that this could happen.* We choose to evaluate unawareness as it addresses perception from a knowledge angle. More precisely, we check if the user knows about a risk prior to being affected by it.
(b) Surprising: score "one (1)" as *Completely unsurprising* to "five (5)" as *Completely surprising.* Moving beyond knowledge, surprisingness gives a sense of the "interestingness" of a risk to a user. Indeed, prior work in highlighting interesting correlations within personal informatics [56] found surprisingness to be a statistically significant ($p < 0.01$) predictor of interestingness to a user.
(c) Severity: score "one (1)" as *Not severe at all (it does not bother you at all)* to "five (5)" as *Extremely severe (you would not use USDT because of this possibility).* Going further than knowledge and interestingness, severity is an action-oriented indication of the user's perceived impact of a risk on their own usage behavior. Severity is also a typical feature of Web3 audit reports, which are a common practice in the smart contract ecosystem that enhance user trust [41, 53].

Similar to the user interface evaluation, we followed each score with a text field asking the respondent to explain their choice. Additionally, we note that the measurements of surprisingness and awareness in **RQ1** are in danger of being undifferentiated by users due to their semantic similarity. To mitigate this, we added additional descriptive text to each question. For surprisingness, the description was "*Does it surprise you that this could happen, or did it happening make sense to you?*". For unawareness, the description was "*Prior to reading this reason, did you know that your transaction could be rejected for this reason?*"

Statistical Analysis: Our analysis includes statistical significance tests using the Mann-Whitney U test for unpaired analysis, and the Wilcoxon signed rank test for paired analysis. We also reported normality and Levene tests, median values, p-values and U-statistics[2]. These tests were performed on the following three pairs of groups of respondents, each rated on a Likert scale (one: *not proficient*, five: *extremely proficient*):
(a) Programmers vs. Non-Programmers: We classified respondents with a self-rated programming proficiency of at least two[3] out of five (85 respondents, 77.3%) as programmers (see **RQ1**, **RQ2**, **RQ4**).
(b) High vs. Low Web3 Proficiency: We classified respondents with a self-rated Web3 proficiency at least four out of five (32 respondents, 29.1%) as high Web3 proficiency (see **RQ1**, **RQ2**, **RQ4**).
(c) High vs. Low Behavior Anticipation of Users: We classified respondents with a self-rated ability to anticipate smart contract behavior of at least four out of five (87 users, 79.1%) as confident in **RQ4**.

4.2 ERC-20 Source Code Analysis (RQ4)

Source Code Collection: We analyzed all Ethereum transactions over 94 days from 22nd March 2022 (block 14434001 to block 15012398). From the available source code on Etherscan [11], we collected the top 500 (out of 11M) recipient addresses by transaction volume, and filtered to ERC-20 contracts (see Figure 3). The USDT contract had the most transactions by far, accounting for over 12.7% of the transactions sent to those top 500 addresses. We thus based our analysis around it as the focal point.

Automated Analysis: We conducted automated analysis of the source code of the remaining (78) ERC-20 contracts for only three risks (*Contract Pause*, *User Blacklist* and *Contract Upgrade*). The *Insufficient Funds* and *Transfer Fee Increase* risks were excluded as they were not amenable to our approach of string matching of function names. The strings used in our automated analysis, for case-insensitive string matching on function names, are "*paus*" for "*Contract Pause*", "*blacklist*" for "*User Blacklist*" and "*deprecat*" for "*Contract Upgrade*". These are based on the main components of the function names used by USDT for the respective features (e.g., we stem *"pause"* to *"paus"* for better generalizability). This methodology fails to detect code semantics e.g., it will not detect the "Contract Pause" feature implemented with a function named *stop* (thus failing the match term *paus*). However, after manually evaluating the results for the examined ERC-20 contracts, the string matching was found effective in practice.

[1]Note that in the user study, we examined the inverse, i.e., "awareness" rather than "unawareness". However, this was inverted in the presentation of results (section 5) for consistent analysis of negative reasons (unawareness)

[2]Statistical analysis data and results are publicly available (*see* section 8).
[3]The programmer threshold differs from the other two thresholds to ensure that each group in the statistical test has ≥ 20 participants - with a threshold of four, there would only be 15 participants in the programmer group.

Our *automated analysis* is implemented in about 1.1 KLOC of Python and JavaScript code. The detection experiment took ≈3 minutes, using a single thread on a 16-inch Macbook Pro (2021 model, M1 Max CPU, 32GB RAM, 1TB SSD).

Manual Evaluation: To validate and evaluate the automated detections, we indepedently conducted *manual analysis* to identify instances of *Insufficient Funds* and *Transfer Fee Increase*, determine additional risks and identify the true/false positives for the automated analysis. To this end, a single researcher inspected all source code to identify presence of risky features. For verification, a second researcher inspected a 10% sample of the source code to identify the same risky features, independent of the first researcher's results. Our data is publicly available (section 8).

5 EVALUATION RESULTS

RQ1 Risk Perception: We evaluate end-users' unawareness, surprisingness and severity of transfer risks when using the USDT smart contract. Given a description of transactions affected by each risk, we asked participants to rate the level of each aforementioned property using Likert scales between one (e.g., least surprising) to five (e.g., most surprising). We also collect and analyze the reasons for participants ratings. This includes both the real transfer risks, and the fictitious (fake) ones (see section 4). Our findings are shown in Figure 4, Table 3 and Table 4.

Real Transfer Risks: We found that *most (71.8%) end-users see blacklisting as the most severe risk*, e.g., because it leads to loss of control over their USDT assets (*see* Table 2). As an example, one respondent gave the reason that "Being able to blacklist somebody from transferring their own cryptocurrency is an extremely severe issue, that is basically the same thing as blocking somebody's bank account with money in it". We also observed that *more than half (55.5% and 51.8%) of end-users are surprised and unaware of contract upgrading*. Qualitative analysis (i.e., coding, *see* Table 2) of free-text responses shows this to be largely due to user unawareness of the contract owner's ability to upgrade contracts (e.g., "I had no idea that this could happen"), to upgrading being an unexpected behavior (e.g., "It's surprising as I assumed there would need to be a solid reason for rejecting the contract, especially after the company making the choice to upgrade it") or to a lack of user notice prior to upgrade (e.g., "They should inform customers of these things so that it doesn't come as a shock to them when it happens"). Pausing is the least severe and surprising because end-users believe it is temporary (e.g., "I dont think this pause is permanent"), infrequent (e.g., "It doesn't bother me unless it's constant ") or happens for legitimate reasons (e.g., "It was just temporary for funds safety").

Insufficient funds and transfer fee increase are considered not risky by most users: most users (81.8% and 60.9%) are aware, not surprised or unconcerned about the severity of both risks (*see* Figure 4, Table 2 and Table 3). Rejection due to insufficient funds is generally seen as commonsensical, with some anchoring on their experience with traditional banking systems (e.g., "It's like a bank so not too sursprising"). Most end-users also believe increases in fees are normal (e.g., "Most companies take a fee for money transfers so this is completely unsurprising"), for profit (e.g., "The company would want to profit from transactions"), or are necessary for maintenance (e.g., "As a result of the system maintenance fees, it is not a serious problem"), and it does not deter their usage of the smart contract, especially if agreed to ahead of time (e.g., "Not a severe issue, that is why it's important to read the terms and conditions.").

These results imply that most end-users believe blacklisting and contract upgrade are highly risky, pausing is somewhat risky, but insufficient funds, and fee increase are low risks.

> *Most (up to 71.8%) end-users believe contract upgrade and user blacklist to be the most severe and surprising transfer risks.*

Real versus Fake Transfer Risks: We note that *users rated fake risks similarly as real risks across all three metrics*, i.e., surprisingness, unawareness and severity (*see* Figure 4, Table 3 and Table 4). For instance, the majority (up to 40%) of users *wrongly* claim to be highly aware (score one) of *fake* risks – receiver's rejecting a transaction or receiving less USDT due to insufficient funds. Notably, this rating is more than their awareness for all *real* risks (except insufficient funds) where the highest awareness scores (score one) are for pausing and fee increase with only 27% and 30.9%, respectively. These numeric ratings, validated by our qualitative analysis of the free-text responses, suggest that *end-users are as surprised, unaware and concerned about the severity of real risks as much as fictitious, fake risks.* Users confusing and similarly rating real and fake risks shows they are uninformed about the risks inherent in USDT transfers.

> *End-users are uninformed (unaware) about real transfer risks. They confuse real and fake risks, rating both as similarly surprising and severe, while being incorrectly more informed (aware) about fake risks.*

Statistical Analysis and Differences in Distributions: We tested for statistically significant differences in risk rating, differentiated by users' self-rated proficiency in programming and Web3 separately (*see* section 4 for setup details). *There was no statistically significant difference between tested groups across all metrics, i.e., unawareness, surprisingness and severity.* This suggests that neither self-rated programming nor high Web3 proficiency are related to risk perception of end-users. Notably, none of the proficiencies yielded better identification of fake risks. We also note that even with both thresholds changed to three, programming and Web3 proficiency still show no statistically significant differences.

Additionally, in order to better intuit the (lack of) differences between the groups, we provide violin plots of risk perception distributions across two different thresholds for both programming and Web3 proficiency skill. These are found in Appendix 8, with Figure 5 serving as a representative example.

> *Self-rated programming/Web3 proficiency does not significantly influence end-users rating of real and fake transfer risks.*

RQ2 Understanding of User Interface: We investigate end-user understanding of the MetaMask (USDT) smart contract user interface in communicating transaction outcomes i.e., *full amount transferred*, a *reduced amount transfered* and *transfer failure* (no amount transferred). We inspect if these outcomes are discoverable and understandable for end-users. Figure 6, Table 5 and Table 6 highlight our results.

Table 2: Reasons for end-users' perception of *real* transfer risks, highest number/percentage of participants and reasons are marked in bold text and *wrong reasons* are in *italicized and bold text* ("#" = "Number of Respondents", "%" = "Percentage of Respondents"). Exclusions for ambiguous/misanswered/incorrect answers and multiply-categorized answers lead to percentage totals and absolute totals deviating from 100% and 110 responses.

Metric	Outcome	Most Prevalent Reasons for Users' Perception of Risks					
		Negative Reasons			Positive Reasons		
		Examples (Why Unaware, Surprising or Severe?)	#	%	**Examples** (Why Aware, Unsurprising or Not Severe?)	#	%
Unaware	**Contract Pause**	**surprising capability** & no personal experience	**40**	**36.4**	maintenance & centralization	14	12.7
	User Blacklist	**no personal experience**, ***it is impossible*** or a bug	**55**	**50**	used to stop bad actors & centralization	40	36.4
	Contract Upgrade	**no knowledge of capability** & unexpected behavior	**57**	**51.8**	prior belief that contract/policy is alterable & sensible	35	31.8
	Transfer Fee Increase	fee is unexpected and unspecified prior to the transfer	41	37.3	**expected financial institution charges** & profit motive	**67**	**60.9**
Surprise	**Contract Pause**	no knowledge of pausing or that it affects all users	41	37.3	**users know of similar incident** & due to centralization	**44**	**40**
	User Blacklist	**no justification provided** & it is unfair	**56**	**50.9**	prior knowledge of capability & it is sensible	37	33.6
	Contract Upgrade	**not aware of capability**, no prior notice and unexpected	**61**	**55.5**	belief capability is possible/anticipated & prior knowledge	31	28.2
	Transfer Fee Increase	less money received, & unaware of capability	23	20.9	**fees are normal**, fees are for profit or maintenance	**36**	**32.7**
Severity	**Contract Pause**	loss of control of assets, impacts all users & centralization	32	29.1	**it is temporary**, infrequent or for reasonable purposes	**63**	**57.3**
	User Blacklist	**loss of control of assets** & not decentralized	**79**	**71.8**	justifiable, benefits all users, & against fraudulent users	16	14.5
	Contract Upgrade	**arbitrary power** & potential for abuse	**44**	**40**	the issue is likely fixable & (only) inconvenient	29	26.4
	Transfer Fee Increase	**users will switch to alternatives**, fee is disliked/unfair	**45**	**40.9**	fee is known prior to transfer & it does not deter usage	41	37.3

Table 3: End-Users' perception of *real* transfer risks. The highest values are marked in bold text ("#" = "Number of Respondents", "%" = "Percentage of Respondents", "Ⓤ" = "Unawareness", "Ⓢ" = "Surprising", "SV" = "Severity")

Score	Contract Pause			User Blacklist			Contract Upgrade			Transfer Fee Increase			Insufficient Funds		
	Ⓤ #/%	Ⓢ #/%	SV #/%	Ⓤ #/%	Ⓢ #/%	SV #/%	Ⓤ #/%	Ⓢ #/%	SV #/%	Ⓤ #/%	Ⓢ #/%	SV #/%	Ⓤ #/%	Ⓢ #/%	SV #/%
1	**27/24.5**	**32/29.1**	17/15.5	22/20.0	21/19.1	4/3.6	12/10.9	22/20.0	21/19.1	**33/30.0**	**39/35.5**	25/22.7	**88/80.0**	**90/81.8**	**85/77.3**
2	23/20.9	16/14.5	11/ 10.0	22/20.0	18/16.4	12/10.9	22/20.0	18/16.4	12/10.9	21/19.1	11/10.0	14/12.7	12/10.9	11/10.0	10/9.1
3	20/18.2	17/15.5	22/20.0	11/10.0	16/14.5	19/17.3	11/10.0	16/14.5	19/17.3	14/12.7	13/11.8	13/11.8	1/0.9	2/1.8	7/6.4
4	19/17.3	19/17.3	26/23.6	21/19.1	23/20.9	38/34.5	21/19.1	23/20.9	**38/34.5**	13/11.8	15/13.6	19/17.3	1/0.9	1/0.9	5/4.5
5	21/19.1	26/23.6	**34/30.9**	**34/30.9**	**32/29.1**	**37/33.6**	**34/30.9**	**32/29.1**	37/33.6	29/26.4	32/29.1	**39/35.5**	8/7.3	6/5.5	3/2.7
Mean	2.85	2.92	**3.45**	3.21	3.25	**3.84**	**3.44**	3.28	3.04	2.85	2.91	**3.3**	1.45	1.38	**1.46**
Std Dev	1.46	1.56	1.42	1.55	1.5	1.12	1.39	1.46	1.41	1.6	1.68	1.6	1.1	1.0	0.99
Median	3	3	**4**	3.5	3.5	**4**	**4**	**4**	3	3	3	**4**	**1**	**1**	**1**

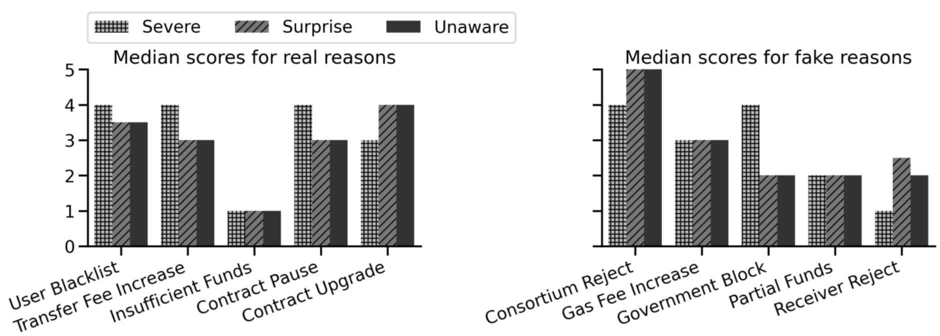


Figure 4: Median scores for rejection and reduction reasons

Risky Transaction Outcomes: Table 5 shows that *end-users find the UI flow of MetaMask to be insufficient for discovering or understanding risky transaction outcomes* (i.e., reduced amount transferred and transfer failure), with reasons highlighted in Table 6. For instance, up to 46.4% (51) users find it difficult to comprehend the "reduced amount" transferred outcome because the flow is misleading (e.g., "The transaction should be 5 usdt plus fees, no reason for it be just 5 usdt"), uninformative (e.g., "It does not give a warning about this") or contradictory to the outcome (e.g., "The transactions actually shows that 5 USDT will be sent"). In contrast, over twice as many (2X) users could comprehend the full amount transferred outcome: 82.7% (91) of users understand the successful outcome, while only 31.8% (35) of users understand the reduced amount transferred outcome[4].

Figure 6 also shows that the full amount transferred flow has a high comprehensibility score (median score of five), while comprehensibility of the other outcomes is low (median scores of two

[4]The actual absolute numbers are provided in Table 5. Note that Table 6 only reports the coded qualitative responses, hence the percentage totals and absolute totals may deviate from 100% and 110 responses due to the exclusion of ambiguous/incorrect answers.

Table 4: End-Users' perception of *fictitious* (fake) transfer risks. The highest values are marked in bold text ("#" = "Number of Respondents", "%" = "Percentage of Respondents", "Ⓤ" = "Unawareness", "Ⓢ" = "Surprising", "ⓈⓋ" = "Severity")

Score	Consortium Reject			Government Block			Receiver Reject			Partial Funds			Gas Fee Increase		
	Ⓤ #/%	Ⓢ #/%	ⓈⓋ #/%	Ⓤ #/%	Ⓢ #/%	ⓈⓋ #/%	Ⓤ #/%	Ⓢ #/%	ⓈⓋ #/%	Ⓤ #/%	Ⓢ #/%	ⓈⓋ #/%	Ⓤ #/%	Ⓢ #/%	ⓈⓋ #/%
1	9/8.2	10/9.1	14/12.7	**32/29.1**	**40/36.4**	12/10.9	**40/36.4**	**44/40.0**	**69/62.7**	**44/40.0**	**50/45.5**	**48/43.6**	31/28.2	**34/30.9**	18/16.4
2	9/8.2	9/8.2	8/7.3	24/21.8	20/18.2	12/10.9	18/16.4	11/10.0	13/11.8	12/10.9	9/8.2	21/19.1	19/17.3	16/14.5	19/17.3
3	13/11.8	10/9.1	15/13.6	13/11.8	13/11.8	16/14.5	12/10.9	14/12.7	17/15.5	12/10.9	11/10.0	15/13.6	15/13.6	16/14.5	20/18.2
4	23/20.9	19/17.3	32/29.1	11/10.0	7/6.4	24/21.8	15/13.6	20/18.2	7/6.4	14/12.7	13/11.8	11/10.0	12/10.9	10/9.1	20/18.2
5	**56/50.9**	**62/56.4**	**41/37.3**	30/27.3	30/27.3	**46/41.8**	25/22.7	21/19.1	4/3.6	28/25.5	27/24.5	15/13.6	**33/30.0**	**34/30.9**	**33/30.0**
Mean	3.98	**4.04**	3.71	2.85	2.7	**3.73**	**2.7**	2.66	1.76	**2.73**	2.62	2.31	2.97	2.95	**3.28**
Std Dev	1.31	1.35	**1.37**	1.6	**1.65**	1.39	**1.61**	1.6	1.15	1.68	**1.7**	1.46	1.62	**1.65**	1.47
Median	**5**	**5**	4	2	2	**4**	2	**2.5**	1	**2**	**2**	**2**	**3**	**3**	**3**

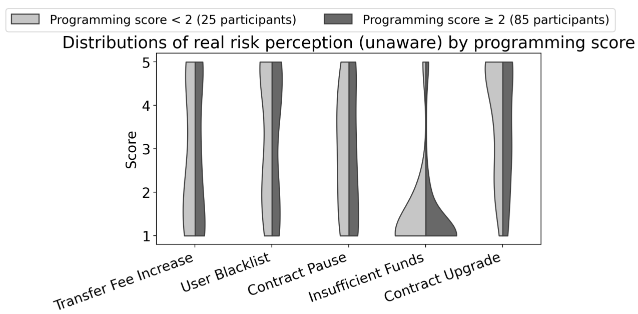


Figure 5: A violin plot comparing distributions of risk perception response scores, segmented by programming skill. Exhaustive plots for the facets of risk perception questions (unawareness, surprisingness, severity), realness (real, fake) and skill level (programming and Web3) at two different thresholds are found in Appendix 8.

Table 5: End-Users' perception of discoverability ("Discov.") and understandability ("Underst.") of USDT transaction outcomes using the MetaMask UI. The highest number/percentage of participants and scores are marked in bold ("#" = "Number of Respondents", "%" = "Percentage of Respondents")

Score	Full amount transferred		Reduced amount transferred		Transfer failure	
	Discov. #/%	Underst. #/%	Discov. #/%	Underst. #/%	Discov. #/%	Underst. #/%
1	11/10.0	5/4.5	**40/36.4**	**36/32.7**	**42/38.2**	**43/39.1**
2	3/2.7	5/4.5	18/16.4	20/18.2	9/8.2	10/9.1
3	7/6.4	9/8.2	15/13.6	19/17.3	14/12.7	18/16.4
4	29/26.4	17/15.5	15/13.6	21/19.1	19/17.3	16/14.5
5	**60/54.5**	**74/67.3**	22/20.0	14/12.7	26/23.6	23/20.9
Mean	4.13	4.36	2.65	2.61	2.8	2.69
Std Dev	1.27	1.11	1.57	1.43	1.65	1.6
Median	5	5	2	2	3	3

and three). As seen in the overview of reasons provided in Table 6, we note that for the transfer failure outcome, this is primarily due to the user flow not explaining the reasons behind the failure (e.g., "There were no prior knowledge shared as to whether an upgrade was being undertaking, also the account had sufficient balance to

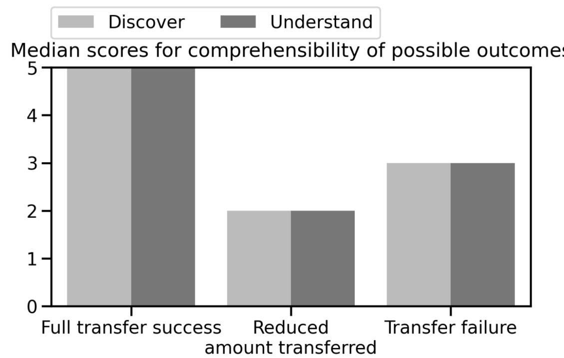


Figure 6: End-user knowledge of smart contracts

ensure that the transaction goes through successfully and finally there were no signs of the account being blocked.")

> *Most (82.7% of) end-users find the MetaMask UI to be sufficient to comprehend the full amount transferred outcome, while fewer users could comprehend the risky transaction outcomes (31.8% and 35.5%).*

Statistical Analysis: We found *no statistically significant differences in end-user ability to discover or understand potential outcomes from the MetaMask UI*, across tested groups of experts (self-rated programmers and Web3 proficiency) versus non-experts. The poor comprehension is thus not due to the lack of self-rated proficiency.

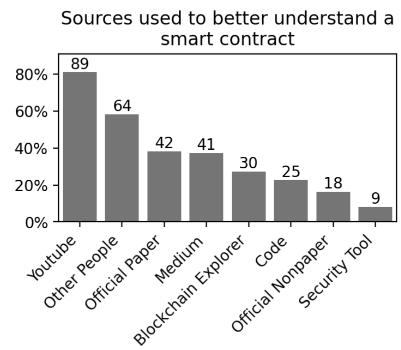
> *Self-rated programming/Web3 proficiency does not affect end-user (in)comprehension of transfer outcomes in the UI.*

RQ3 Smart Contract Understanding: In order to investigate how end-users understand smart contracts, we examine sources of information they use (*see* Figure 7a, Figure 7b). In particular, we ask participants to provide the sources of information they use to educate themselves about the behavior of smart contracts they interact with. We also ask them to rate their perceived levels of knowledge before and after using a smart contract (*see* Figure 8), their trust that smart contracts will behave as they expect (*see* Figure 9a) and their perceived ability to anticipate the behavior of a smart contract (*see* Figure 9b).

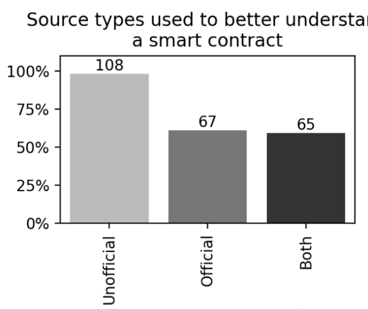
Sources of Information: *All respondents employ external sources, besides the smart contract wallet UI, to educate themselves on the*

Table 6: Reasons for end-users' perception of discoverability ("Disc.") and understandability ("Underst.") USDT transaction outcomes using the MetaMask UI. The highest number/percentage of participants and reasons are marked in bold text and *wrong reasons* are in *italicized and bold text* ("#" = "Number of Respondents", "%" = "Percentage of Respondents"). Exclusions for ambiguous/misanswered/incorrect answers and multiply-categorized answers lead to percentage totals and absolute totals deviating from 100% and 110 responses.

		Most Prevalent Reasons for Users' Perception of Transaction Outcomes					
		Negative Reasons			**Positive Reasons**		
Metric	**Outcome**	**Examples** (Why *Not* Discoverable/Understandable?)	#	%	**Examples** (Why Discoverable/Understandable?)	#	%
Disc.	**Full**	N/A (no meaningful subcategories emerged)	2	1.8	**understandable flow**, & displayed information	**87**	**79.1**
	Reduced	**misleading/unexplained user flow** & unexpected/ ***impossible fee***	**50**	**45.5**	fee is expected & the UI flow is understandable	21	19.1
	Failure	**contradictory flow** & no failure reason	**47**	**42.7**	failure report after occurrence & no failure cost	34	30.9
Underst.	**Full**	more details needed in the user flow	9	8.2	**sufficient funds** & user flow is explanatory	**92**	**83.6**
	Reduced	**user flow is uninformative, contradictory**, or unexpected	**51**	**46.4**	expected fee & contract parameters knowledge	14	12.7
	Failure	**no failure reasons** & no indication of possibility,	**50**	**45.5**	possibly blacklisted & parameters not met	24	21.8



(a) Information sources for users



(b) Type of information source

Figure 7: End-users' information sources

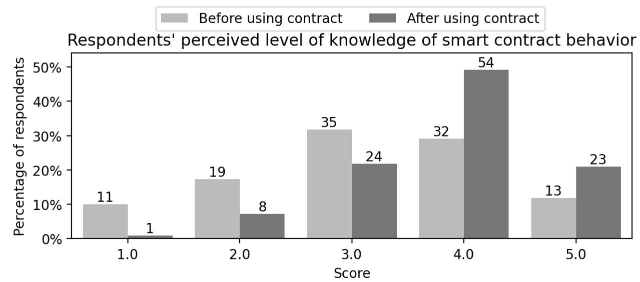


Figure 8: End-user knowledge of smart contracts

behavior of the smart contract. Figure 7b shows that while almost all (98.2%, 108) respondents used an unofficial source of information (e.g., YouTube videos (*see* Figure 7a)), under two-thirds (60.9%, 67) of respondents employ an official source (e.g., source code or whitepaper). Overall, 59.1% (65) of respondents employ both official and unofficial information sources. This implies that end-users prefer unofficial sources to learn about smart contracts. Thus, we recommend that unofficial modes (e.g., videos and blogposts) are also used to communicate smart contract implementation.

Additionally, respondents with relevant self-rated proficiencies used official sources more frequently: 52% of non-programmers, compared to 63.5% of programmers. Likewise, 52.6% of users without high Web3 proficiency used official sources, compared to 79.4% of users with Web3 proficiency.

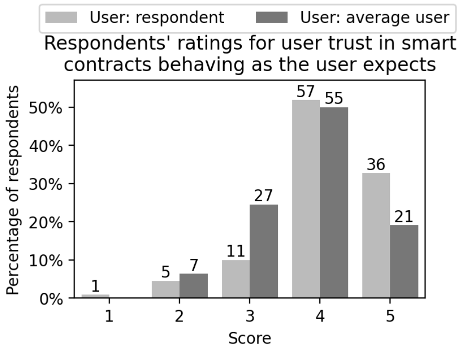
> *Most (98.2% of) end-users employ unofficial sources of information (e.g., Youtube) for self-education on smart contract behaviors.*

Level of Knowledge: We found that *respondents are (19%) more informed about a smart contract's behavior after using the contract versus before using it, on average (3.8 vs. 3.2 mean scores).* Figure 8 shows the level of informedness (least to most informed) of users based on a five-point Likert scale. The difference in the knowledge level *before* and *after* contract usage is statistically significant (W = 384, p = 6.33E-8 under the Wilcoxon signed rank test). This suggests that users become more knowledgeable through experience.

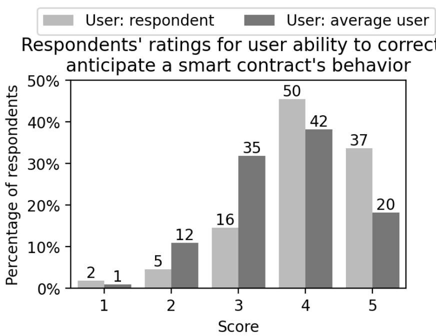
Statistical Analysis (Level of Knowledge): We analyzed the effects of self-rated Web3 and programming proficiency (separately) on levels of knowledge before and after smart contract use. *There were no statistically significant differences for either proficiency.* Users in general seem to benefit from experiences with the smart contract.

Trust: We found that *users trusted their own expectation of smart contract behavior (8%) more than they believed the average smart contract user would trust hers, on average (mean 4.1 versus 3.8, see Figure 9a).* This difference is statistically significant (W = 547, p = 0.0022 under the Wilcoxon signed rank test). Additionally, self-rated programming or high Web3 proficiency had no statistically significant effect on these scores. This suggests that none of the proficiencies influences end-user trust in smart contract behavior.

Behavior Anticipation: *Users believe they are able to anticipate smart contract's behavior 11% better than the average user (mean 4.0*



(a) Trust in Smart Contracts



(b) Anticipate Contract Behavior

Figure 9: End-user trust and anticipation in smart contract behavior, rating both themselves and their perceptions of what they see as the average user.

vs. 3.6 in Figure 9b). This difference was also statistically significant (W = 593.5, p = 1.9E-4 under Wilcoxon's signed rank test) and Figure 9b shows that the difference holds across most scores. Similar to trust scores, self-rated programming proficiency and high Web3 proficiency had no statistically significant effect. This suggests that neither programming nor Web3 proficiency influence users' confidence in their ability to anticipate smart contract behaviors.

> *Most end-users believe they are (up to 11%) more anticipatory and trusting of smart contracts behaviors than the average user.*

Statistical Analysis (Behavior Anticipation and Risk Perception): *End-users' belief in their ability to anticipate smart contract behavior does not imply their awareness of USDT transfer risks.* There was no significant difference in high and low behavior anticipation users for USDT transfer risks.

> *A high self-assessment in anticipating smart contract behavior does not imply that a user is more aware of USDT transfer risks than other users.*

RQ4 Generalizability of Risks: We extracted all known ERC-20 contracts (78, excluding USDT) out of the top 500 Ethereum recipient addresses (see Figure 3). We analyzed their source code for (a) the existence/prevalence of the five (5) real transfer risks examined in **RQ1** as well as the existence of a transfer fee, (b) the existence and prevalence of other (new) risks, and (c) the effectiveness of our experimental string-matching heuristic in automatically detecting three of the five risks in these 78 contracts (see Table 7 and Table 8).

Table 7: Prevalence of Risky Features in the top ERC-20 contracts. ("#" = "Number of Contracts", "P": Risk identified *prior* to analysis, "D": Risk identified *during* analysis)

Feature	#	%	Example	P / D
Insufficient Funds	78	100%	MaticToken	P
Contract Pause	15	19.2%	MANAToken	P
User Blacklist	1	1.3%	KOKContract	P
Contract Upgrade	1	1.3%	AVINOCToken	P
Transfer Fee Increase	1	1.3%	Shibnobi	P
Transfer Fee	4	5.1%	SaitamaInu	P
Arbitrary Mint	21	26.9%	Stronger	D
Transfer Limit Change	2	2.6%	KishuInu	D
Destroy User Funds	1	1.3%	SmartToken	D

Prevalence of Risks: We found occurrences of all (five) transfer risks across examined smart contracts (*see* Table 7). Some of the least prevalent risks are the most severe and surprising for users. In particular, contract upgrade and user blacklist are the least prevalent (1.3%) but rated most severe and surprising in **RQ1**. In contrast, the most prevalent transfer risk (contract pause: 19.2%) is the least severe and least surprising transfer risk (*see* **RQ1**). Overall, these results imply the need to expose and clearly explain rare risks to end-users.

> *The (five) transfer risks in USDT are present in the examined top ERC-20 contracts and up to 19.2% prevalent.*

Other Risks: *In addition to the examined transfer risks, we discovered three (3) other risky and unexpected features with potentially significant impact* (see Table 7). These risks are either beyond a user's ability to transfer tokens, or else not present in the USDT contract. *Arbitrary Mint* allows one or more authorized addresses to arbitrarily add new tokens to the total token supply. This may be misused for price manipulation via supply increase. *Transfer Limit Change* allows an authorized address to limit the amount of tokens transferred by other users per transaction. This potentially limits users from selling tokens during time-sensitive periods, making them more vulnerable to Pump and Dump schemes [89]. *Destroy(ing) User Funds* allows an authorized address to destroy any particular user's token holdings without justification. These risks may reduce trust and understanding of contracts as they are not exposed via the UI. These results motivate the need to further study users' risks in smart contract usage.

> *We found three additional risks in ERC-20 smart contracts with potentially significant impact on end-user understanding and trustworthiness.*

Effectiveness of Automated Detection: Table 8 shows that *our proposed risk detection approach is effective especially in detecting Contract Pause.* We recall that our detection is based on partial string matching of function names (*see* section 4). *Contract Pause* detection (F1=80%) performed well, but *User Blacklist* (F1=100%) and *Contract Upgrade* (F1 not applicable) tests were only evaluated on one positive case. Overall, this result shows that the automatic detection of transactions risks is feasible. It also motivates developing more effective methods, e.g., using program analysis techniques.

> *Our automated risk detection was effective in detecting two out of three attempted transfer risks.*

6 DISCUSSION AND FUTURE WORK

In sections 6.1 and 6.2, we address takeaways which are relevant to both end-users and designers of smart contracts and wallet interfaces. In section 6.3, we detail some potential directions for future research, centered around addressing the core issue of end-user explainability of smart contracts. The majority of these directions are relevant for designers.

6.1 User Education and Risk Perception

Insufficient information sources: *While all respondents in the study attempted to educate themselves on smart contracts they use, they still lacked sufficient risk comprehension* (*see* **RQ3**). We observed that most resources focus on communicating the goals of the respective project, but they do not provide the concrete implementation of the smart contract, which dictates the rules of interaction with end-users. For instance, the official Tether whitepaper [5] made no reference to blacklisting, pausing and contract upgrade. Additionally, previous work [34] investigating inconsistent token behavior noted that of 752 whitepapers of inconsistent tokens, only 31 (4.1%) included detailed token behavior descriptions. Some developer-facing resources make transfer risk features explicit, e.g., OpenZeppelin [13] and Alchemy [15]. However, these resources do not target end-users (e.g., non-developers).

To improve the users' comprehension of smart contracts, we recommend the source code should be made publicly available. At present, only bytecode is available on the Ethereum blockchain and not all projects upload their source code for public inspection. Most (430 of the top 500) recipient addresses have bytecode available, but 62 (14.4%) of those do not have verified source code on Etherscan. End-users desire to understand the source code of the smart contracts: Almost one in every four (19 /85 = 22.4%) respondents with programming ability read smart contract source code (**RQ3**).

Confidence/skills vs. Risk comprehension: *Neither self-rated programming/Web3 proficiency nor high ability to anticipate smart contract behavior have statistical significance on risk comprehension (see* ***RQ1****).* We thus suggest that many users may have an inflated sense of confidence, as 79.1% (87) users rated themselves as having high behavior anticipation ability yet generally performed badly in risk awareness. Similarly, users do no seem to be effectively engaging relevant skills (self-rated programming ability, Web3 proficiency) to better comprehend these risks (*see* statistical tests in **RQ1**). We believe further research is needed to uncover the reasons behind inflated self-confidence, and why users may ineffectively use relevant skills for smart contract comprehension.

6.2 Common Erroneous Beliefs

Tether can communicate directly to users: *Many respondents erroneously believed that Tether would be able to communicate changes to all affected users.* For instance, regarding the upgrading capability, one user believed that "*...the company Tether Limited was supposed to inform me about upgrading the smart contract*". The ability for Tether to directly notify a user by their MetaMask wallet does not exist in the USDT contract. Respondents, however, frequently implied otherwise. For example, 15 (13.6%) respondents noted that a fee increase would only be surprising if no notice was given. Besides, 12 (10.9%) respondents claimed that the blacklisting was surprising due to no reason given for being blacklisted. We note it is not possible, at present, to communicate such reasons via the wallet interface. We thus expect that measuring and improving user comprehension of smart contract is a fertile ground for future research.

Tether is decentrally governed: *In the absence of appropriate tutorial, users may believe that blockchain properties, such as decentralization, transfer over to the projects held on the blockchain.* For instance, 58 (52.7%) study respondents incorrectly believed that USDT is "governed in a decentralized manner". Of the 58 (52.7%) respondents who themselves owned USDT, 28 (48.3%) held the same incorrect belief. Alternatively, many users seem to anchor expectations on their experiences with centralized institutions. For example, 19 (17.3%) respondents rated themselves as aware of a potential USDT fee increase as "*companies/financial institutions increasing transaction charges is normal*". Interestingly, this anchoring in some of our participants is consistent with the "bank bias" found for non-users in prior work [60]. We expect that tutorial is necessary for smart contract comprehension.

6.3 Future Research Directions in Explainable Smart Contracts

This work demonstrates the *explainability gap* between smart contract end-users and designers. In particular, this gap refers to how end-users poorly understand smart contracts, despite using them. As such, we lay out the following research directions for potential future work in both understanding and bridging this gap.

The explainability gap in other smart contracts. *More work is needed to confirm that this gap is present for other smart contracts, beyond the ERC-20 smart contracts.* Due to the standardization of the ERC-20 specification, we were able to generalize our insights from USDT. However, some heavily-used smart contract are not as standardized or well-specified as the ERC-20 smart contracts (e.g., the MakerDAO Vat contract [22]). Thus, there is a need to investigate end-user understanding of such heterogenous smart contracts.

Understanding end-users and factors which affect their understanding. *Smart contract understanding is not homogeneous among end-users. Hence, more research is needed to determine the level of user understanding and factors influencing understanding level.* As discussed, our results reveal a tension wherein some users incorrectly attributed centralized properties to USDT while others incorrectly attributed decentralized properties to it. As an example, explaining their complete lack of surprise at the pausing feature,

Table 8: Effectiveness of automated transfer risk detection

Feature	Accuracy	F1 score	Precision	Recall	True Positive	False Positive	True Negative	False Negative
Contract Pause	0.92	0.80	0.80	0.80	12	3	60	3
User Blacklist	1	1	1	1	1	0	77	0
Contract Upgrade	0.99	N/A	N/A	0	0	0	77	1

one user incorrectly transfers their experience on a centralized offchain exchange to the Tether contract ("I have witnessed it before with binance."). On the other side, explaining their full surprise at the same feature, another user mentioned that they "thought this was supposed to be decentralized". Reinforcing this divide, 58 participants (52.7%) incorrectly answered that USDT is decentrally governed when asked. This is consistent with prior work [60], although our focus is on actual end-users. In addition to these two incorrect attributions, some participants demonstrate correct understanding. Again explaining a complete lack of surprise at the same feature, a different user correctly claims that "Tether Limited controls the issuance and management of USDT, they have the ability to pause the smart contract that governs the stablecoin, which would prevent any transactions from being processed.". In follow-up work, we would examine the distinct factors causing different levels of end-user understanding. Above shedding descriptive light on the space of end-users, such work may also help designers to take informed decisions in order to increase overall end-user understanding.

Increasing explainability in light wallet user interfaces. *Through identification of common patterns in source code, wallet interfaces might be able to present additional salient information to the end-user.* As seen through both our work and TokenScope [34], a significant portion of ERC-20 contracts extend or deviate from the specification in common ways. While this deviation is observable in the source code, we note that even among our participants with programming experience, more than three quarters (77.6%) have never read any smart contract source code. Thus, a potential research direction is increasing the scope of explanations downstream in the wallet interfaces. While our study is focused on MetaMask due to its overwhelming market dominance, we also inspected two other high-usage wallets (Trust Wallet [19] and OKX Wallet [23], each with one million users in the Chrome Web Store [20, 21]) through the same YUSDT-based procedure (*see* subsection 4.1). All three wallets failed to explain these transfer risks in a manner similar to that shown in Figure 2. In **RQ4**, we show that a simple string matching approach is effective at detecting some of these patterns, which suggests that more sophisticated approaches may achieve even greater success.

Increasing explainability of smart contract source code. *Enhancing the explainability of source code may facilitate end user understanding.* Smart contract programming languages are generally Turing-complete, and this expressiveness vastly hinders the ability to derive explanations from them. However, the source code is the root of the smart contract and therefore any improvements in explainability here is likely to make explainability easier downstream (e.g., in the UI). To this end, a pattern-based approach for identifying common patterns and extracting explanations from them might be useful. Such an approach has indeed been used in the context of security to identify administrative patterns [54] and to aid in loop summarization [62]. An alternative approach is to modify the smart contract programming language. While smart contract programming languages have emerged both in industry and academia [80], few of them focus on enhancing the explainability of the source code to end-users.

7 LIMITATIONS AND THREATS TO VALIDITY

We note three limitations regarding our user study. First, while our user study is done with a large number (110) of respondents with varying demographics (e.g., industries and countries), it was focused on the USD Tether smart contract and recruited users solely from the Prolific platform. Additionally, users from some countries may not be able to access Google Forms, which we used for our survey. As discussed in subsection 6.3, more work is needed to generalize our findings to other users and smart contracts. Second, both our transfer risk evaluation metrics (surprisingness, awareness and severity) and UI evaluation metrics (discoverability and understandability) are measured through self-assessment rather than an investigation of actual user behavior. Thus, our findings may differ from behavior-oriented studies (e.g. , observational study). Mitigating these threats, we employed attention-checking, validation and knowledge checking questions. We also conducted pilot studies to revise confusing questions, add follow-up questions, and identify users' misunderstanding or contradictory responses. Third, we note some internal inconsistencies in participant response: e.g., five participants claimed to simultaneously own no stablecoins and yet own USDT, despite being informed prior that USDT is a stablecoin. We conjecture that this is caused by participants failing to make the connection between their experience and new knowledge provided in the survey.

Regarding our source code analysis (**RQ4**), results may not generalize to other periods (e.g., before 2022), or non-ERC-20 smart contracts. However, we note that during the period of our search, ERC-20 implementations were the most used standard (25.6% of transaction volume) for the top recipient addresses with published source code (368 addresses out of the top 500). To further mitigate this threat, we have provided our experimental data.

8 CONCLUSION

This paper investigates end-users' comprehension of smart contract transfer risks using the most popular smart contract (USDT), a widely used smart contract interface (MetaMask) and 78 frequently used ERC-20 contracts. We observed that respondents are unaware of transfer risks, irrespective of their self-rated programming ability or Web3 proficiency. Users also consider the current

USDT/MetaMask UI flow to be insufficient in communicating transaction outcomes. We further analyzed the 78 next most frequently used ERC-20 contracts, after USDT, to show that the transfer risks considered in our study are prevalent beyond USDT. This analysis also discovered additional transfer risks beyond the transfer risks considered in our study. Our research points to the need for *explainable smart contracts*. Additionally, we hope that this work will motivate policy-makers to make informed decisions regarding *end-user understanding* of smart contracts. This work demonstrates that significant research is required in user interface design to explain the risky transaction outcomes to smart contract users. We hope this work provides a foundation for further research in improving end-user comprehension of smart contract transfer risks. For reproduction and further research, our research data and code are available in the following:

https://zenodo.org/communities/tether-study

REFERENCES

[1] 2015. ERC-20 Token Standard. https://eips.ethereum.org/EIPS/eip-20. Accessed: 2023-07-18.

[2] 2018. ERC-1155 Multi Token Standard. https://eips.ethereum.org/EIPS/eip-1155. Accessed: 2023-07-30.

[3] 2018. ERC-721 Non-Fungible Token Standard. https://eips.ethereum.org/EIPS/eip-721 Accessed: 2023-07-30.

[4] 2021. Polygon: Ethereum's Internet of Blockchain. https://api-new.whitepaper.io/documents/pdf?id=rk1I25PzO. Accessed: 2023-07-31.

[5] 2022. Tether: Fiat currencies on the Bitcoin blockchain. https://whitepaper.io/document/6/tether-whitepaper. Accessed: 2023-07-28.

[6] 2023. All Chains TVL - DefiLlama. https://defillama.com/chains. Accessed: 2023-07-30.

[7] 2023. BNB Smart Chain. https://github.com/bnb-chain/whitepaper/blob/master/WHITEPAPER.md. Accessed: 2023-07-30.

[8] 2023. The crypto wallet for Defi, Web3 Dapps and NFTs | MetaMask. https://metamask.io/. Accessed: 2023-07-30.

[9] 2023. Ether Market Capitalization Chart. http://etherscan.io/chart/marketcap. Accessed: 2023-07-21.

[10] 2023. Ethereum Unique Addresses Chart. http://etherscan.io/chart/address. Accessed: 2023-07-21.

[11] 2023. Etherscan: Verify and Publish Contract Source Code. https://etherscan.io/verifyContract. Accessed: 2023-07-28.

[12] 2023. Google Forms. https://docs.google.com/forms/. Accessed: 2023-07-28.

[13] 2023. OpenZeppelin | Contracts. https://www.openzeppelin.com/contracts. Accessed: 2023-10-30.

[14] 2023. Prolific. https://www.prolific.co/. Accessed: 2023-07-28.

[15] 2023. Smart Contract Repository. https://www.alchemy.com/smart-contracts. Accessed: 2023-10-30.

[16] 2023. Tether USDT price today. https://coinmarketcap.com/currencies/tether/. Accessed: 2023-07-21.

[17] 2023. Total Value Locked (TVL). https://coinmarketcap.com/alexandria/glossary/total-value-locked-tvl. Accessed: 2023-07-31.

[18] 2023. What is the Sepolia testnet? https://www.alchemy.com/overviews/sepolia-testnet. Accessed: 2023-07-29.

[19] 2024. Best Crypto Wallet for Web3, NFTs and DeFi | Trust — trustwallet.com. https://trustwallet.com/. Accessed 12-07-2024.

[20] 2024. OKX Wallet - Chrome Web Store — chromewebstore.google.com. https://chromewebstore.google.com/detail/okx-wallet/mcohilncbfahbmgdjkbpemcciiolgcge. Accessed 12-07-2024.

[21] 2024. Trust Wallet - Chrome Web Store — chromewebstore.google.com. https://chromewebstore.google.com/detail/trust-wallet/egjidjbpglichdcondbcbdnbeeppgdph. Accessed 12-07-2024.

[22] 2024. Vat - Detailed Documentation. https://docs.makerdao.com/smart-contract-modules/core-module/vat-detailed-documentation. Accessed: 2024-12-09.

[23] 2024. Web3 Crypto Wallet | DeFi Wallet | NFT Wallet — okx.com. https://www.okx.com/web3/rewritethesystem. Accessed 12-07-2024.

[24] 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 2021 CHI Conference on Human Factors in Computing Systems (CHI '21)*. ACM, 1–19. https://doi.org/10.1145/3411764.3445679

[25] Ghada Almashaqbeh, Allison Bishop, and Justin Cappos. 2019. ABC: A Cryptocurrency-Focused Threat Modeling Framework. *CoRR* abs/1903.03422 (2019). arXiv:1903.03422 http://arxiv.org/abs/1903.03422

[26] Christian Bräm, Marco Eilers, Peter Müller, Robin Sierra, and Alexander J. Summers. 2021. Rich specifications for Ethereum smart contract verification. *Proc. ACM Program. Lang.* 5, OOPSLA (2021), 1–30.

[27] Clemens Brunner, Günther Eibl, Peter Fröhlich, Andreas Sackl, and Dominik Engel. 2021. Who Stores the Private Key? An Exploratory Study about User Preferences of Key Management for Blockchain-based Applications. In *ICISSP*. SCITEPRESS, 23–32.

[28] Vitalik Buterin et al. 2014. A next-generation smart contract and decentralized application platform. *white paper* 3, 37 (2014), 2–1.

[29] Stefanos Chaliasos, Marcos Antonios Charalambous, Liyi Zhou, Rafaila Galanopoulou, Arthur Gervais, Dimitris Mitropoulos, and Ben Livshits. 2024. Smart Contract and DeFi Security Tools: Do They Meet the Needs of Practitioners?. In *Proceedings of the 46th IEEE/ACM International Conference on Software Engineering*. 1–13. https://doi.org/10.1145/3597503.3623302 arXiv:2304.02981 [cs].

[30] Kathy Charmaz. 2006. *Constructing grounded theory: A practical guide through qualitative analysis*. sage.

[31] Haoxian Chen, Lan Lu, Brendan Massey, Yuepeng Wang, and Boon Thau Loo. 2024. Verifying Declarative Smart Contracts. In *Proceedings of the 46th IEEE/ACM International Conference on Software Engineering*.

[32] Jiachi Chen, Xin Xia, David Lo, and John C. Grundy. 2022. Why Do Smart Contracts Self-Destruct? Investigating the Selfdestruct Function on Ethereum. *ACM Trans. Softw. Eng. Methodol.* 31, 2 (2022), 30:1–30:37.

[33] Jiachi Chen, Xin Xia, David Lo, John C. Grundy, Xiapu Luo, and Ting Chen. 2022. Defining Smart Contract Defects on Ethereum. *IEEE Trans. Software Eng.* 48, 2 (2022), 327–345.

[34] Ting Chen, Yufei Zhang, Zihao Li, Xiapu Luo, Ting Wang, Rong Cao, Xiuzhuo Xiao, and Xiaosong Zhang. 2019. TokenScope: Automatically Detecting Inconsistent Behaviors of Cryptocurrency Tokens in Ethereum. (Nov. 2019), 1503–1520.

[35] Yuichiro Chinen, Naoto Yanai, Jason Paul Cruz, and Shingo Okamura. 2021. RA: A Static Analysis Tool for Analyzing Re-Entrancy Attacks in Ethereum Smart Contracts. *J. Inf. Process.* 29 (2021), 537–547.

[36] Hanting Chu, Pengcheng Zhang, Hai Dong, Yan Xiao, Shunhui Ji, and Wenrui Li. 2023. A survey on smart contract vulnerabilities: Data sources, detection and repair. *Inf. Softw. Technol.* 159 (2023), 107221.

[37] Michael J. Coblenz, Reed Oei, Tyler Etzel, Paulette Koronkevich, Miles Baker, Yannick Bloem, Brad A. Myers, Joshua Sunshine, and Jonathan Aldrich. 2020. Obsidian: Typestate and Assets for Safer Blockchain Programming. *ACM Trans. Program. Lang. Syst.* 42, 3 (2020), 14:1–14:82.

[38] Michael J. Coblenz, Joshua Sunshine, Jonathan Aldrich, and Brad A. Myers. 2019. Smarter smart contract development tools. In *WETSEB@ICSE*. IEEE / ACM, 48–51.

[39] Philip Daian, Steven Goldfeder, Tyler Kell, Yunqi Li, Xueyuan Zhao, Iddo Bentov, Lorenz Breidenbach, and Ari Juels. 2020. Flash Boys 2.0: Frontrunning in Decentralized Exchanges, Miner Extractable Value, and Consensus Instability. In *IEEE Symposium on Security and Privacy*. IEEE, 910–927.

[40] Xun Deng, Sidi Mohamed Beillahi, Cyrus Minwalla, Han Du, Andreas Veneris, and Fan Long. 2024. Safeguarding DeFi Smart Contracts against Oracle Deviations. http://arxiv.org/abs/2401.06044 arXiv:2401.06044 [cs].

[41] Ding Feng, Rupert Hitsch, Kaihua Qin, Arthur Gervais, Roger Wattenhofer, Yaxing Yao, and Ye Wang. 2023. *DeFi Auditing: Mechanisms, Effectiveness, and User Perceptions*. Springer Nature Switzerland, 320–336. https://doi.org/10.1007/978-3-031-48806-1_21

[42] Michael Froehlich, Philipp Hulm, and Florian Alt. 2021. Under Pressure. A User-Centered Threat Model for Cryptocurrency Owners. In *2021 4th International Conference on Blockchain Technology and Applications (ICBTA 2021)*. ACM, 39–50. https://doi.org/10.1145/3510487.3510494

[43] Michael Froehlich, Maurizio Raphael Wagenhaus, Albrecht Schmidt, and Florian Alt. 2021. Don't Stop Me Now! Exploring Challenges Of First-Time Cryptocurrency Users. In *Designing Interactive Systems Conference 2021 (DIS '21)*. ACM. https://doi.org/10.1145/3461778.3462071

[44] Michael Fröhlich, Felix Gutjahr, and Florian Alt. 2020. Don't lose your coin! Investigating Security Practices of Cryptocurrency Users. In *Proceedings of the 2020 ACM Designing Interactive Systems Conference (DIS '20)*. ACM, 1751–1763. https://doi.org/10.1145/3357236.3395535

[45] Michael Fröhlich, Franz Waltenberger, Ludwig Trotter, Florian Alt, and Albrecht Schmidt. 2022. Blockchain and Cryptocurrency in Human Computer Interaction: A Systematic Literature Review and Research Agenda. In *Conference on Designing Interactive Systems*. ACM, 155–177.

[46] Ulrich Gallersdörfer, Jonas Ebel, and Florian Matthes. 2021. Augmenting MetaMask to Support TLS-endorsed Smart Contracts. In *DPM/CBT@ESORICS (Lecture Notes in Computer Science, Vol. 13140)*. Springer, 227–244.

[47] Xianyi Gao, Gradeigh D. Clark, and Janne Lindqvist. 2016. Of Two Minds, Multiple Addresses, and One Ledger: Characterizing Opinions, Knowledge, and Perceptions of Bitcoin Across Users and Non-Users. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI'16)*. ACM, 1656–1668. https://doi.org/10.1145/2858036.2858049

[48] Zhipeng Gao, Lingxiao Jiang, Xin Xia, David Lo, and John Grundy. 2021. Checking Smart Contracts With Structural Code Embedding. *IEEE Trans. Software Eng.* 47, 12 (2021), 2874–2891.

[49] Maggie Yongqi Guan, Yaman Yu, Tanusree Sharma, Molly Zhuangtong Huang, Kaihua Qin, Yang Wang, and Kanye Ye Wang. 2024. Security Perceptions of Users in Stablecoins: Advantages and Risks within the Cryptocurrency Ecosystem. Cryptology ePrint Archive, Paper 2024/1538. https://eprint.iacr.org/2024/1538

[50] 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)*.

[51] Xing Hu, Zhipeng Gao, Xin Xia, David Lo, and Xiaohu Yang. 2021. Automating User Notice Generation for Smart Contract Functions. In *ASE*. IEEE, 5–17.

[52] Mingyuan Huang, Jiachi Chen, Zigui Jiang, and Zibin Zheng. 2024. Revealing Hidden Threats: An Empirical Study of Library Misuse in Smart Contracts. In *Proceedings of the 46th IEEE/ACM International Conference on Software Engineering*. ACM, Lisbon Portugal, 1–12. https://doi.org/10.1145/3597503.3623335

[53] Molly Zhuangtong Huang, Rui Jiang, Tanusree Sharma, and Kanye Ye Wang. 2024. Exploring User Perceptions of Security Auditing in the Web3 Ecosystem. *Cryptology ePrint Archive* (2024).

[54] Nikolay Ivanov, Hanqing Guo, and Qiben Yan. 2021. Rectifying Administrated ERC20 Tokens. In *Information and Communications Security*, Debin Gao, Qi Li, Xiaohong Guan, and Xiaofeng Liao (Eds.). Springer International Publishing, Cham, 22–37.

[55] Hyeji Jang and Sung H. Han. 2022. User experience framework for understanding user experience in blockchain services. *Int. J. Hum. Comput. Stud.* 158 (2022), 102733.

[56] Simon L. Jones and Ryan Kelly. 2016. Finding "Interesting" Correlations in Multi-Faceted Personal Informatics Systems. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI EA '16)*. Association for Computing Machinery, New York, NY, USA, 3099–3106. https://doi.org/10.1145/2851581.2892401

[57] Irni Eliana Khairuddin and Corina Sas. 2019. An Exploration of Bitcoin Mining Practices: Miners' Trust Challenges and Motivations. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3290605.3300859

[58] John Kolb, Moustafa AbdelBaky, Randy H. Katz, and David E. Culler. 2021. Core Concepts, Challenges, and Future Directions in Blockchain: A Centralized Tutorial. *ACM Comput. Surv.* 53, 1 (2021), 9:1–9:39.

[59] Ye Liu, Yi Li, Shang-Wei Lin, and Rong Zhao. 2020. Towards automated verification of smart contract fairness. In *ESEC/SIGSOFT FSE*. ACM, 666–677.

[60] Alexandra Mai, Katharina Pfeffer, Matthias Gusenbauer, Edgar R. Weippl, and Katharina Krombholz. 2020. User Mental Models of Cryptocurrency Systems - A Grounded Theory Approach. In *SOUPS @ USENIX Security Symposium*. USENIX Association, 341–358.

[61] Easwar Vivek Mangipudi, Udit Desai, Mohsen Minaei, Mainack Mondal, and Aniket Kate. 2023. Uncovering Impact of Mental Models towards Adoption of Multi-device Crypto-Wallets. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security (CCS '23)*. ACM, 3153–3167. https://doi.org/10.1145/3576915.3623218

[62] Benjamin Mariano, Yanju Chen, Yu Feng, Shuvendu K. Lahiri, and Isil Dillig. 2021. Demystifying loops in smart contracts. In *Proceedings of the 35th IEEE/ACM International Conference on Automated Software Engineering* (Virtual Event, Australia) *(ASE '20)*. Association for Computing Machinery, New York, NY, USA, 262–274. https://doi.org/10.1145/3324884.3416626

[63] Mark Mossberg, Felipe Manzano, Eric Hennenfent, Alex Groce, Gustavo Grieco, Josselin Feist, Trent Brunson, and Artem Dinaburg. 2019. Manticore: A User-Friendly Symbolic Execution Framework for Binaries and Smart Contracts. In *ASE*. IEEE, 1186–1189.

[64] Dave Murray-Rust, Chris Elsden, Bettina Nissen, Ella Tallyn, Larissa Pschetz, and Chris Speed. 2022. Blockchain and Beyond: Understanding Blockchains Through Prototypes and Public Engagement. *ACM Trans. Comput. Hum. Interact.* 29, 5 (2022), 41:1–41:73.

[65] Say Keat Ooi, Chai Aun Ooi, Jasmine AL Yeap, and Tok Hao Goh. 2021. Embracing Bitcoin: users' perceived security and trust. *Quality & Quantity* 55 (2021), 1219–1237.

[66] Daniel Perez and Benjamin Livshits. 2021. Smart Contract Vulnerabilities: Vulnerable Does Not Imply Exploited. In *USENIX Security Symposium*. USENIX Association, 1325–1341.

[67] Benedikt Putz, Manfred Vielberth, and Günther Pernul. 2022. BISCUIT - Blockchain Security Incident Reporting based on Human Observations. In *ARES*. ACM, 27:1–27:6.

[68] Meng Ren, Fuchen Ma, Zijing Yin, Ying Fu, Huizhong Li, Wanli Chang, and Yu Jiang. 2021. Making smart contract development more secure and easier. In *ESEC/SIGSOFT FSE*. ACM, 1360–1370.

[69] Jorge Saldivar, Elena Martínez-Vicente, David Rozas, Maria-Cruz Valiente, and Samer Hassan. 2023. Blockchain (not) for Everyone: Design Challenges of Blockchain-based Applications. In *Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems (CHI '23)*. ACM, 1–8. https://doi.org/10.1145/3544549.3585825

[70] Corina Sas and Irni Eliana Khairuddin. 2017. Design for Trust: An Exploration of the Challenges and Opportunities of Bitcoin Users. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM, 6499–6510. https://doi.org/10.1145/3025453.3025886

[71] Ilya Sergey, Vaivaswatha Nagaraj, Jacob Johannsen, Amrit Kumar, Anton Trunov, and Ken Chan Guan Hao. 2019. Safer smart contract programming with Scilla. *Proc. ACM Program. Lang.* 3, OOPSLA (2019), 185:1–185:30.

[72] Chaochen Shi, Borui Cai, Yao Zhao, Longxiang Gao, Keshav Sood, and Yong Xiang. 2023. CoSS: Leveraging Statement Semantics for Code Summarization. *IEEE Trans. Software Eng.* 49, 6 (2023), 3472–3486.

[73] Chaochen Shi, Yong Xiang, Jiangshan Yu, Keshav Sood, and Longxiang Gao. 2023. Machine translation-based fine-grained comments generation for solidity smart contracts. *Inf. Softw. Technol.* 153 (2023), 107065.

[74] Janice Jianing Si, Tanusree Sharma, and Kanye Ye Wang. 2024. Understanding User-Perceived Security Risks and Mitigation Strategies in the Web3 Ecosystem. In *Proceedings of the CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '24)*. Association for Computing Machinery, New York, NY, USA, Article 974, 22 pages. https://doi.org/10.1145/3613904.3642291

[75] Sunbeom So, Myungho Lee, Jisu Park, Heejo Lee, and Hakjoo Oh. 2020. VERISMART: A Highly Precise Safety Verifier for Ethereum Smart Contracts. In *IEEE Symposium on Security and Privacy*. IEEE, 1678–1694.

[76] Yuqiang Sun, Daoyuan Wu, Yue Xue, Han Liu, Haijun Wang, Zhengzi Xu, Xiaofei Xie, and Yang Liu. 2023. GPTScan: Detecting Logic Vulnerabilities in Smart Contracts by Combining GPT with Program Analysis. http://arxiv.org/abs/2308.03314 arXiv:2308.03314 [cs].

[77] Nick Szabo. 1997. Formalizing and securing relationships on public networks. *First Monday* 2, 9 (Sept. 1997). https://doi.org/10.5210/fm.v2i9.548

[78] Christof Ferreira Torres, Mathis Baden, Robert Norvill, Beltran Borja Fiz Pontiveros, Hugo Jonker, and Sjouke Mauw. 2020. ÆGIS: Shielding Vulnerable Smart Contracts Against Attacks. In *AsiaCCS*. ACM, 584–597.

[79] Christof Ferreira Torres, Julian Schütte, and Radu State. 2018. Osiris: Hunting for Integer Bugs in Ethereum Smart Contracts. In *ACSAC*. ACM, 664–676.

[80] Ángel Jesús Varela-Vaca and Antonia M. Reina Quintero. 2022. Smart Contract Languages: A Multivocal Mapping Study. *ACM Comput. Surv.* 54, 1 (2022), 3:1–3:38.

[81] Emanuele Viglianisi, Mariano Ceccato, and Paolo Tonella. 2020. A federated society of bots for smart contract testing. *J. Syst. Softw.* 168 (2020), 110647. https://api.semanticscholar.org/CorpusID:219432640

[82] Artemij Voskobojnikov, Borke Obada-Obieh, Yue Huang, and Konstantin Beznosov. 2020. Surviving the cryptojungle: Perception and management of risk among North American cryptocurrency (non) users. In *International conference on financial cryptography and data security*. Springer, 595–614.

[83] Artemij Voskobojnikov, Oliver Wiese, Masoud Mehrabi Koushki, Volker Roth, and Konstantin (Kosta) Beznosov. 2021. The U in Crypto Stands for Usable: An Empirical Study of User Experience with Mobile Cryptocurrency Wallets. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems (CHI '21)*. ACM, 1–14. https://doi.org/10.1145/3411764.3445407

[84] Zhiyuan Wan, Xin Xia, David Lo, Jiachi Chen, Xiapu Luo, and Xiaohu Yang. 2021. Smart Contract Security: a Practitioners' Perspective. In *ICSE*. IEEE, 1410–1422.

[85] Mingyue Wang, Yu Guo, Chen Zhang, Cong Wang, Hejiao Huang, and Xiaohua Jia. 2023. MedShare: A Privacy-Preserving Medical Data Sharing System by Using Blockchain. *IEEE Trans. Serv. Comput.* 16, 1 (2023), 438–451.

[86] Ye Wang, Patrick Zuest, Yaxing Yao, Zhicong Lu, and Roger Wattenhofer. 2022. Impact and User Perception of Sandwich Attacks in the DeFi Ecosystem. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems* (New Orleans, LA, USA) *(CHI '22)*. Association for Computing Machinery, New York, NY, USA, Article 591, 15 pages. https://doi.org/10.1145/3491102.3517585

[87] Xiaolin Wen, Kim Siang Yeo, Yong Wang, Ling Cheng, Feida Zhu, and Min Zhu. 2023. Code Will Tell: Visual Identification of Ponzi Schemes on Ethereum. In *CHI Extended Abstracts*. ACM, 70:1–70:6.

[88] Yunpeng Xiao, Bufan Deng, Siqi Chen, Kyrie Zhixuan Zhou, Ray LC, Luyao Zhang, and Xin Tong. 2024. "Centralized or Decentralized?": Concerns and Value Judgments of Stakeholders in the Non-Fungible Tokens (NFTs) Market. *Proceedings of the ACM on Human-Computer Interaction* 8, CSCW1 (April 2024), 1–34. https://doi.org/10.1145/3637305

[89] Jiahua Xu and Benjamin Livshits. 2019. The anatomy of a cryptocurrency {Pump-and-Dump} scheme. In *28th USENIX Security Symposium (USENIX Security 19)*. 1609–1625.

[90] Tingting Yin, Shuohan Wu, zihao li, Luyi Yan, Weimin Chen, Muhui Jiang, Chenxu Wang, Xiapu Luo, and Hao Zhou. 2024. Revealing Hidden Threats: An Empirical Study of Library Misuse in Smart Contracts. In *Proceedings of the 46th IEEE/ACM International Conference on Software Engineering*. ACM, 1–12.

[91] Tingting Yin, Chao Zhang, Yuandong Ni, Yixiong Wu, Taiyu Wong, Xiapu Luo, Zheming Li, and Yu Guo. 2022. An Empirical Study on Implicit Constraints in Smart Contract Static Analysis. In *ICSE (SEIP)*. IEEE, 31–32.

[92] Xiao Liang Yu, Omar I. Al-Bataineh, David Lo, and Abhik Roychoudhury. 2020. Smart Contract Repair. *ACM Trans. Softw. Eng. Methodol.* 29, 4 (2020), 27:1–27:32.

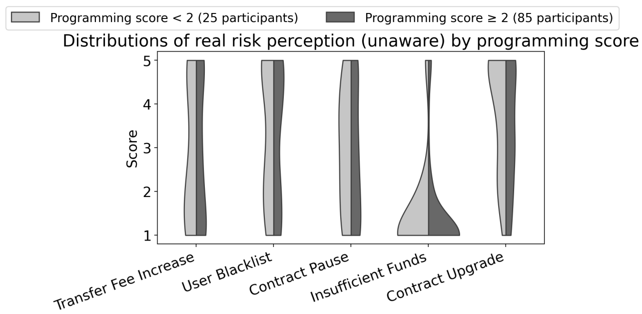
[93] Yaman Yu, Tanusree Sharma, Sauvik Das, and Yang Wang. 2024. "Don't put all your eggs in one basket": How Cryptocurrency Users Choose and Secure Their Wallets. In *Proceedings of the CHI Conference on Human Factors in Computing Systems (CHI '24)*. ACM, 1–17. https://doi.org/10.1145/3613904.3642534

[94] Yuyao Zhang, Siqi Ma, Juanru Li, Kailai Li, Surya Nepal, and Dawu Gu. 2020. SMARTSHIELD: Automatic Smart Contract Protection Made Easy. In *SANER*. IEEE, 23–34.

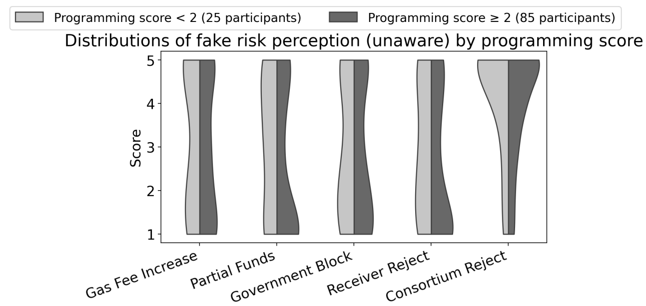
[95] Liyi Zhou, Xihan Xiong, Jens Ernstberger, Stefanos Chaliasos, Zhipeng Wang, Ye Wang, Kaihua Qin, Roger Wattenhofer, Dawn Song, and Arthur Gervais. 2023. SoK: Decentralized Finance (DeFi) Attacks. In *2023 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2444–2461. https://doi.org/10.1109/sp46215.2023.10179435

[96] Chenguang Zhu, Ye Liu, Xiuheng Wu, and Yi Li. 2022. Identifying Solidity Smart Contract API Documentation Errors. In *ASE*. ACM, 56:1–56:13.

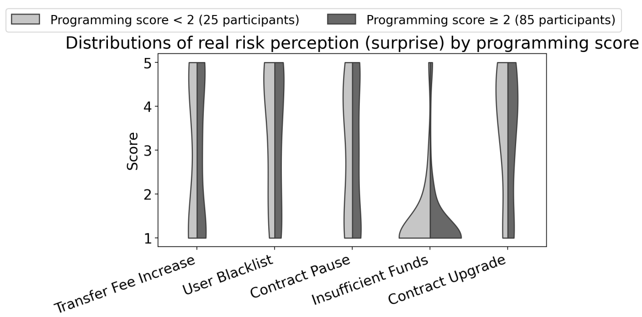
A APPENDIX



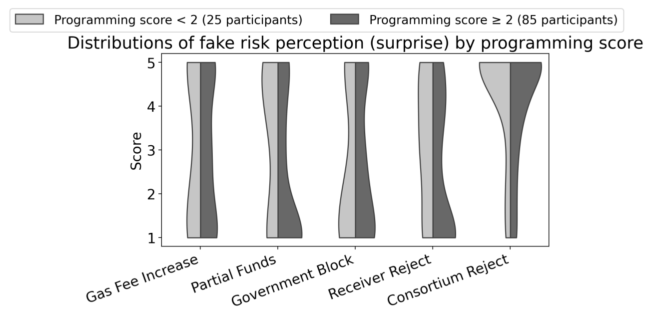
(a) Unawareness for real risks



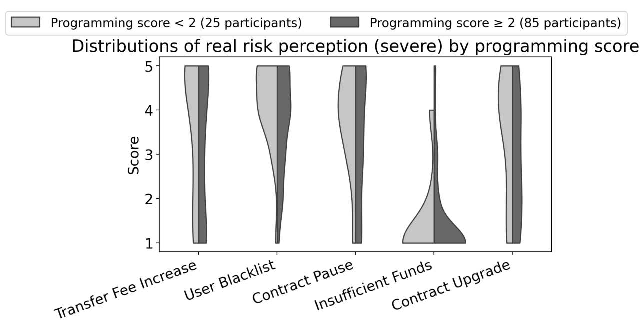
(b) Unawareness for fake risks



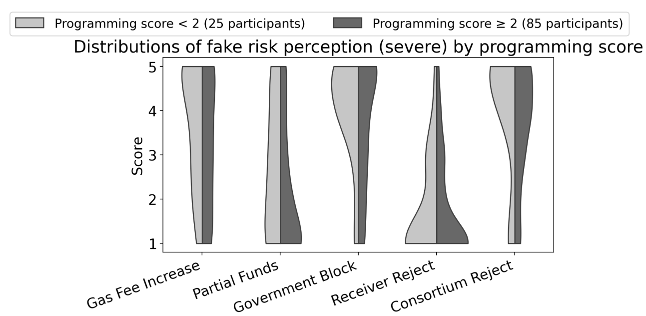
(c) Surprisingness for real risks



(d) Surprisingness for fake risks

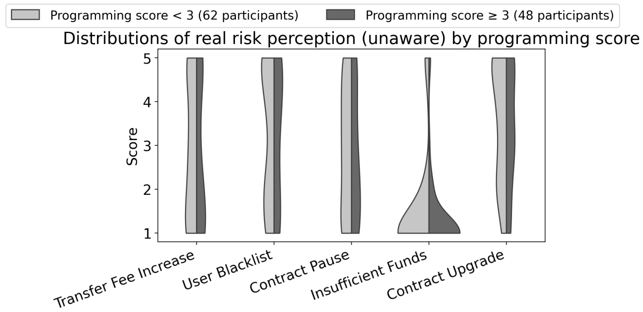


(e) Severity for real risks

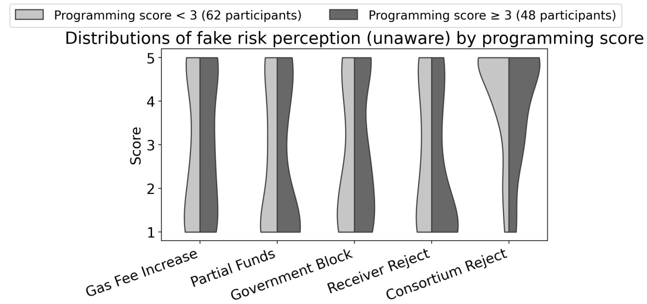


(f) Severity for fake risks

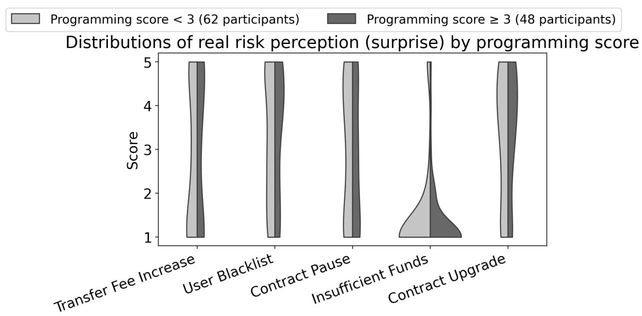
Figure 10:
Risk perception distributions split by programming proficiency, with the self-rated skill threshold set to two out of five.



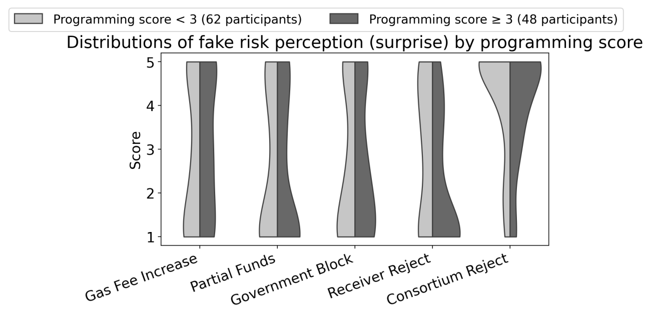
(a) Unawareness for real risks



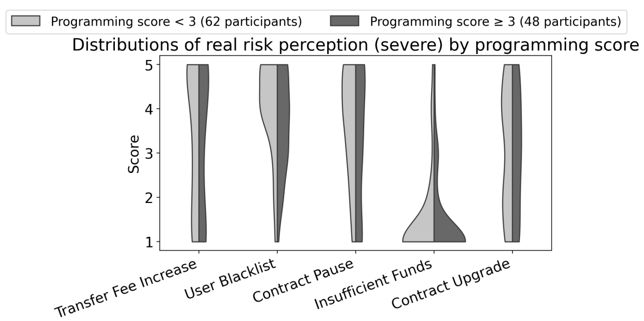
(b) Unawareness for fake risks



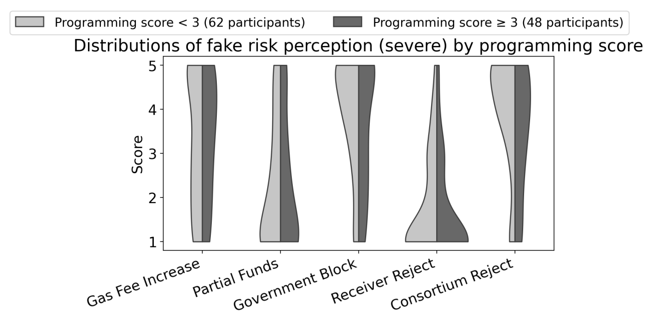
(c) Surprisingness for real risks



(d) Surprisingness for fake risks

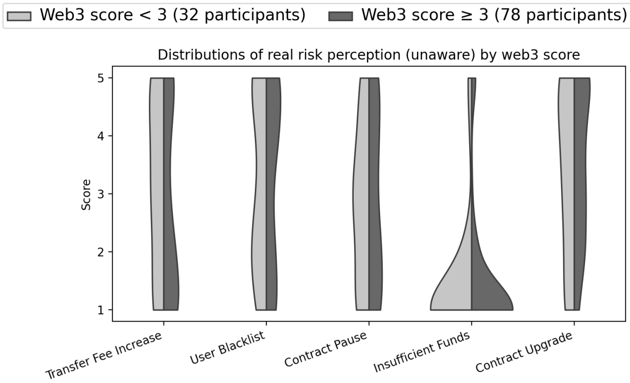


(e) Severity for real risks

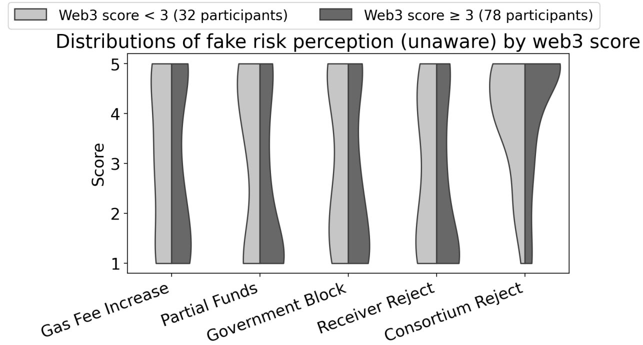


(f) Severity for fake risks

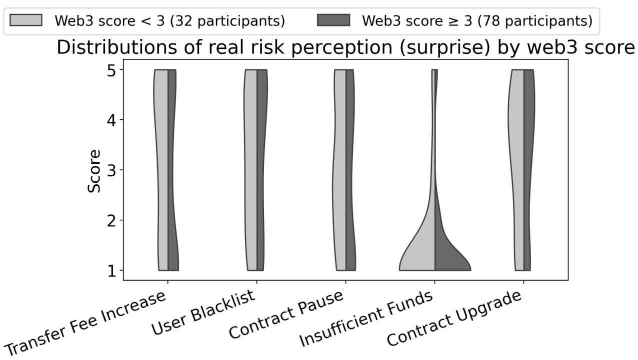
Figure 11:
Risk perception distributions split by programming proficiency, with the self-rated skill threshold set to three out of five.



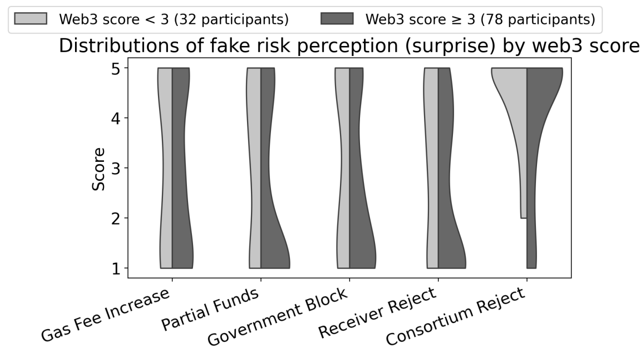
(a) Unawareness for real risks



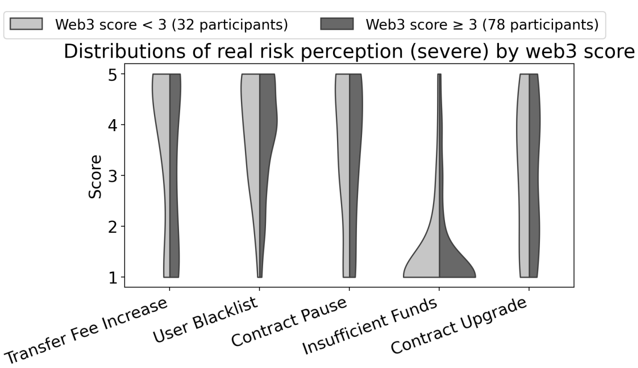
(b) Unawareness for fake risks



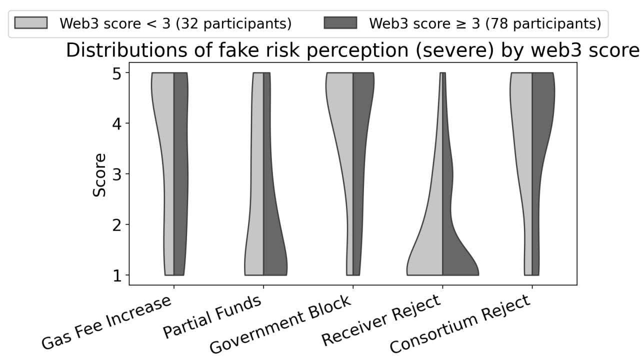
(c) Surprisingness for real risks



(d) Surprisingness for fake risks

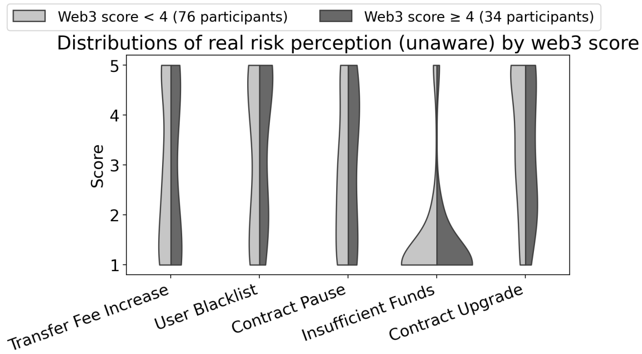


(e) Severity for real risks

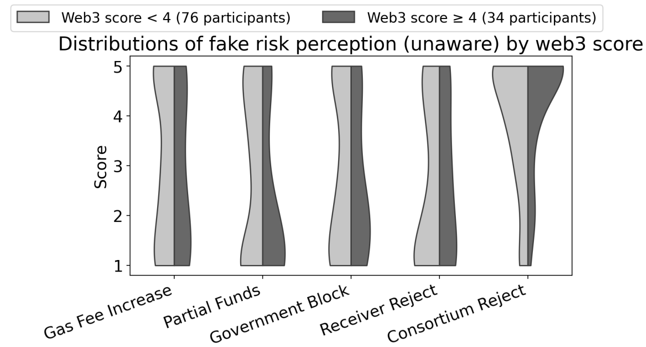


(f) Severity for fake risks

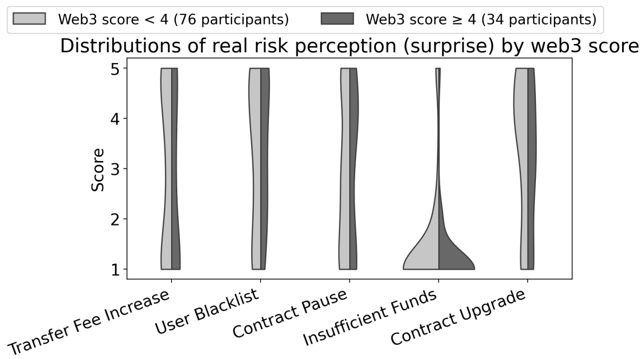
Figure 12:
Risk perception distributions split by Web3 proficiency, with the self-rated skill threshold set to three out of five.



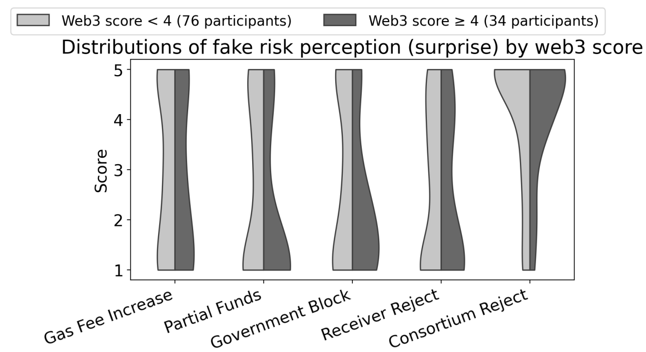
(a) Unawareness for real risks



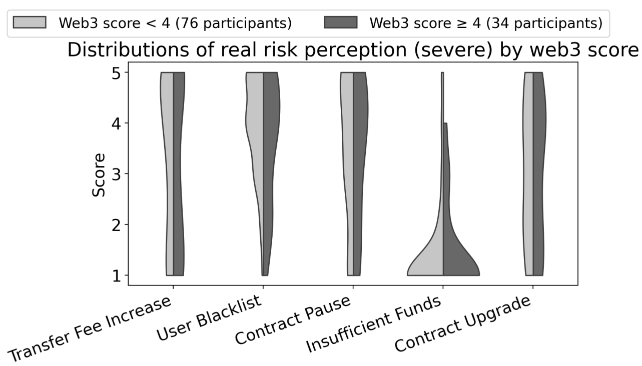
(b) Unawareness for fake risks



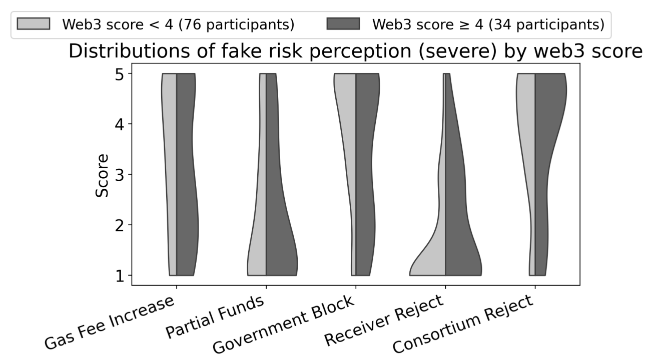
(c) Surprisingness for real risks



(d) Surprisingness for fake risks



(e) Severity for real risks



(f) Severity for fake risks

Figure 13:
Risk perception distributions split by Web3 proficiency, with the self-rated skill threshold set to four out of five.