TEEvil: Identity Lease via Trusted Execution Environments

Ivan Puddu Daniele Lain Moritz Schneider

Elizaveta Tretiakova Sinisa Matetic Srdjan Čapkun

ETH Zurich

Abstract

We investigate *identity lease*, a new type of service in which users lease their identities to third parties by providing them with full or restricted access to their online accounts or credentials. We discuss how identity lease could be abused to subvert the digital society, facilitating the spread of fake news and subverting electronic voting by enabling the sale of votes. We show that the emergence of Trusted Execution Environments and anonymous cryptocurrencies, for the first time, allows the implementation of such a lease service while guaranteeing fairness, plausible deniability and anonymity, therefore shielding its users and renters from prosecution. To show that such a service can be practically implemented, we build an example system that we call TEEVIL leveraging Intel SGX and ZCash. Finally, we discuss defense mechanisms and challenges in the mitigation of identity lease services.

1 Introduction

Different online platforms collect, share and monetize user's data. From this data they infer user behaviors and preferences [67] which can then be used for better product placement or to exert influence over the users [25]. Not only data, but also user actions can have value and be monetized. Third parties can instruct real users to perform arbitrary actions online in exchange for money – an activity called *crowdturfing* [85]. Crowdturfing platforms facilitate these exchanges, by connecting *customers*, entities that want some action to be conducted online to *workers*, regular users that can be hired to perform various tasks. These tasks range from posting advertisements, comments or reviews related to products and venues, interacting with (e.g., "liking") the content on Online Social Networks (OSNs), to click-fraud operations against Pay-per-Click advertisement providers. This type of advertisement is more impactful than the traditional means because it poses as "grass-root" word-of-mouth opinions of (allegedly) real users, whose identity lends credibility to the content [14, 35].

In this work we investigate a new type of user monetization, that we name *identity lease*, in which users lease their accounts and online credentials to third parties. Those third parties can then control those accounts within some defined temporal and other limits.

We show that identity lease has the potential to have a significant societal impact – it can be used to generate and enhance the spread of fake news, and ultimately to sway elections by e.g., buying votes. Existing crowdturfing platforms limit the reach of such activities. Such platforms require *workers* to manually execute tasks and *customers* to subsequently verify them. They do not offer any privacy guarantees for the workers, and when executing payments expose the platform and workers to legal risks – e.g., sale and purchase of votes is illegal in almost all jurisdictions [39, 75, 78].

To have a higher impact, such platforms either need to recruit more workers or resort to the use of fake accounts. As the example of fake news shows, the use of fake accounts can only have a limited impact [72, 76]. The fake news phenomenon is driven by targeted, carefully crafted misinformation. OSNs are then used to maximize the misinformation's impact by allowing to easily target separate audiences with different alternative facts, resulting sometimes in the change of election results [10, 72, 81]. To make such campaigns more effective these stories should not only be started by dubious sources and spread through fake accounts but need to be accepted and spread by real users within their social networks. A party that could lease a large number of real identities would therefore have the ability to spread fake news more effectively and in much more subtle ways. This would amount to the purchase of a large number of small-scale influencers.

Big social networks have been under pressure from governments to address the spread of fake news. Their natural reaction has been to track down fake news sources and fake accounts thus stopping their spread into the networks of real users [34, 76, 77]. However, if fake news were to be introduced by large numbers of real active users and *inserted among their regular posts* then not only would their impact be more pronounced, but it would make their identification and removal

more complex for OSN operators. The identification of such carefully placed fake news would then require looking into message content [21, 73] and their removal would have implications on free speech and censorship [68].

Furthermore, if identity lease would be done without the manual participation of the users while preserving the confidentiality of their OSN credentials and remunerated through anonymous payments, such a system would attract a wider set of workers.

If the platform could in addition guarantee fairness, it would then further offer all the necessary properties for the large scale sale and purchase of votes in an e-voting system. It would ensure that voters get paid only if the buyer gets valid voting credentials, as well as that the payment is executed when the vote is cast by the buyer. It would enable the voters to sell their votes while shielding them from prosecution.

We aim to show that Trusted Execution Environments (TEEs) coupled with distributed ledgers (and thus cryptocurrencies) represent almost an ideal set of technologies that can underpin identity lease. Ultimately, these new technologies create the ability to sell/buy/rent digital identities on a massive scale while preserving indistinguishability, plausible deniability and fairness between customers and worker, therefore, having the potential to undermine entire societies. To illustrate this, we introduce TEEVIL, a system that allows users to lease their online accounts to *renters* while preserving the above properties. Further, our system can provide its users reasonable anonymity guarantees, with the assumption that some specific actions they might sell can potentially de-anonymize them.We implemented TEEVIL within Intel SGX [23] and used ZCash [70] for anonymous payment; the renter leased a Reddit account from the user, allowing it to interact in the user's name.

This paper focuses on the technical design of an identity lease system and discusses countermeasures that could be raised against such systems, all in the hope that such behaviors would be detected and prevented before they emerge in the wild.

Contributions.

Our contributions can be summarized as follows:

- *Problem*: we highlight the problem that arises when a large number of people are allowed to lease their online identities and sell access to their accounts, automatically and without any repercussion. Additionally, we explore the impact of this phenomenon on existing and future e-voting systems [2].

- TEEVIL *System*: We present and implement TEEVIL, an architecture for identity lease that is fair, privacy preserving, and completely automated for account owners and renters. TEEVIL leverages a careful combination of escrows to cryptocurrencies with trusted execution to achieve fairness, and disincentivize all parties from trying to cheat the protocol. TEEVIL protects users credentials by limiting their use through TEEs [59], and protects anonymity of such transactions through anonymous cryptocurrencies [70]. In particular, TEEVIL allows the replacement of the crowdturfing middlemen with TEEs[1], hence removing entities from the picture that can be coerced to reveal information about owners and renters.

- *Countermeasures*: We discuss possible countermeasures against identity lease in general, and to TEEVIL in particular, highlighting that stopping these systems might be challenging in some scenarios and requires more research attention.

Outline. The remainder of the paper is structured as follows: In Section 2, we introduce the problem, describing some applications in which an identity leasing marketplace can be disruptive in current digital societies. In Section 3, we introduce TEEVIL, a protocol for fair and secure identity sharing, and detail its protocol, followed by the security analysis in Section 4. Then, we change perspective and summarize possible defences that can mitigate the effects of TEEVIL in Section 5. In Section 6, we describe our prototype implementation of TEEVIL and analyze its performance. In Section 7, we discuss two improved distributed designs of TEEVIL and how it can be combined with anonymous networks. Finally, we conclude in Section 8.

2 Problem Statement

In this section we introduce the problem through two motivating examples, discuss why existing techniques were not well suited to solve this problem and why TEEs and anonymous cryptocurrencies represent almost ideal candidate technologies in this problem space.

2.1 Motivating examples

Providing the ability to entities to lease identities on a large scale could have a number of negative consequences for digital societies. It can be used to subvert digital societies by polarizing and influencing the opinions of its members, to pollute automatic recommender systems, bootstrap the virality of content on online social networks, and subvert democratic processes, to name a few.

We discuss two examples in more detail: electronic voting and posts in online social networks.

Electronic Voting. Lease of government-issued electronic identities allows miscreants to buy votes in e-voting platforms. While in-person e-vote selling is unpractical (the miscreant would need to be co-located with the seller at the time of

[1] Note that this, however, inherently shifts trust to the TEE manufacturer.

voting, as e-voting systems do not provide any proof of whom a vote was cast for), online rental would allow the miscreant to cast votes to their candidate of choice. There usually exists a subset of people which argue that their vote would not matter, and could, therefore, be persuaded to sell their vote. Even a small amount of votes can swing election results [64] and greatly influence politics and democracies.

Online Social Networks (OSNs). In the context of OSNs, miscreants could leverage peer trust, and post advertisement masked as legitimate content – for example post, "like", or "reshare" some product. Influential peers that maximize the impact of content can be selected with targeted, topic-specific heuristics [7, 17, 41, 52]. This could destroy the conventional centralized advertisement business model of OSNs, shifting to a new peer-to-peer paradigm where every account is a potential "influencer" since users are highly influenced by posts from their peers [71, 91]. Miscreants can also manipulate content virality processes: patterns of content diffusion and virality processes on online communities [18, 29, 38] can be leveraged to maximize the efficacy of the crucial initial phase of the life-cycle of news and rumors [83] – a worrisome scenario for the uncontrolled spreading of fake news. Regarding politics, friendship relationships and trust could be leveraged to "water down" opposite views (for example by commenting or posting that an opposite party politician *"maybe has a point"*). Such a strategy could be impactful, as diverging opinions are otherwise very unlikely to reach "echo chambers" that are oppositely polarized [26]. Rented accounts could also give a fake sense of grass-root approval to extreme views[2], and efficiently promote fake news and conspiracy theories that find fertile soil in already polarized communities due to confirmation biases [9, 83] – and debunking news after they have gone viral is largely ineffective [38, 93].

2.2 Requirements

A system that allows identity lease at a large scale is likely to violate the terms of service of online platforms, and in some cases will be illegal to operate or to participate in, notably in the case of a sale of votes in government elections. To achieve its goal the system therefore needs to include proper monetary incentives but at the same time guarantees fairness and impunity. Such a system would therefore need to satisfy the following properties.

Fairness. Identity owners and identity renter should exchange services for payment. Upon the completion of the transaction, the identity renter would have used the identity owners's account, and the identity owner would have received the payment. There would be no incentive to put up accounts for rental on a system where fairness is not guaranteed; the same holds if identity renters are not guaranteed to receive what they pay for.

Indistinguishability. *From the point of view of the service* any action coming from rented identities should be indistinguishable from actions manually performed by the original identity owners. Given the grey area of identity lease, it is likely that targeted services (for which accounts are put up for rental) would want to block such activities. If they could therefore distinguish between "normal" user actions and actions performed while the account is rented to a third party, they could intervene. Indistinguishability guarantees that the service is oblivious to the rental process.

Plausible Deniability. All parties should be able to plausibly deny their participation in such a system or in a particular campaign. This would make it difficult for services or authorities to take action against identity owners and identity renters.

2.3 Existing Approaches and Limitations

The above mentioned properties are well studied in the security literature. Candidate technologies that could be used by crowdturfing platforms include a range of fair exchange, zero-knowledge, multi-party computation and distributed ledger techniques. However, all these techniques fail in solving this problem and to satisfy the required properties. In fact, most would require the cooperation of online services that are being exploited.

In the example of electronic voting, a fair exchange would require the identity owner to either give its voting credentials to the identity renter or to present the proof that he/she voted in a particular way. Deployed electronic systems do not provide such proofs (e.g., [2]). Furthermore, without a trusted third party, no protocol will guarantee fairness of this exchange [33, 36]. Although multi-party computation and Zero Knowledge (ZK) protocols could be used to achieve some of the required properties, they would all be limited by the interface of the service (e.g., e-voting) to which the identity owner presents its credentials. These services typically offer only simple login credentials and do not run ZK or MPC protocols. Smart contracts [15] that run on top of distributed ledgers can also seem like a good solution to this problem. They were introduced with the Ethereum Virtual Machine, allowing users to run programs on the distributed ledger which can transfer coins to users or other contracts. The execution of the smart contracts is always correct since honest nodes do not accept incorrect execution traces. Recent research has shown, how smart contracts can be used to facilitate fair exchange protocols [30]. However, all inputs, as well as the smart contract itself, must be public. Identity owner credentials would therefore equally be made public making it difficult to ensure fairness or plausible deniability.

Therefore, since existing techniques cannot satisfy all the

[2]The virality of the "PizzaGate" and "QAnon" conspiracy theories was bootstrapped by accounts posing as real people [37].

properties mentioned above at the same time, so far it was impossible to deploy a system that allows leasing identities *en masse* and hence subvert digital societies.

3 TEEVIL

Recently, mainstream vendors, such as Intel [23], ARM [4], and RISC-V [1,24], started deploying Trusted Execution Environments (TEEs) in their processors. TEEs are becoming a commodity technology, increasingly widely deployed even in consumer devices. Due to their diffusion, TEEs are enabling new use-cases thanks to their security properties. However, TEEs can not only be used for good – nefarious applications have yet to be explored. In this paper, we show how TEEs enable building a large-scale marketplace for identity lease.

There are five main parties participating in TEEVIL:

- **Identity Owner**: Any entity that enrolls in TEEVIL to offer some actions to be performed on their behalf.
- **Identity Renter**: Any entity that wants to start an identity rental campaign. The identity renter needs to specify what should be performed through the leased identities, and how much he would pay for such service.
- **Service**: The service is an online social network (OSN), e-voting system, or any online service for which a identity renter is interested in using other people's identities through their accounts to perform some actions on their behalf.
- **TEEVIL Enclave:** The code running in the TEE is responsible for managing all the interactions between the identity owners, the identity renters, the service(s) and the funds in the cryptocurrency. A set of supported functions for the services is exposed to identity renters. Renters can then specify through an API which actions they would like to buy and for which services. The enclave performs these actions on behalf of identity owners and distributes the funds accordingly.
- **Infrastructure Maintainer**: The infrastructure maintainer provides the infrastructure that hosts the enclaves. There could be multiple infrastructure maintainers running the same services, and identity renters and identity owners are free to choose the ones they prefer, based for instance on the fees the infrastructure maintainers charge or the services they support.

TEEVIL replaces the trusted intermediary of crowdturfing campaigns with a protocol that runs on top of TEE and anonymous cryptocurrencies on distributed ledgers. As we show, this combination of a ledger and TEEs enables large-scale identity leasing while guaranteeing fairness and plausible deniability.

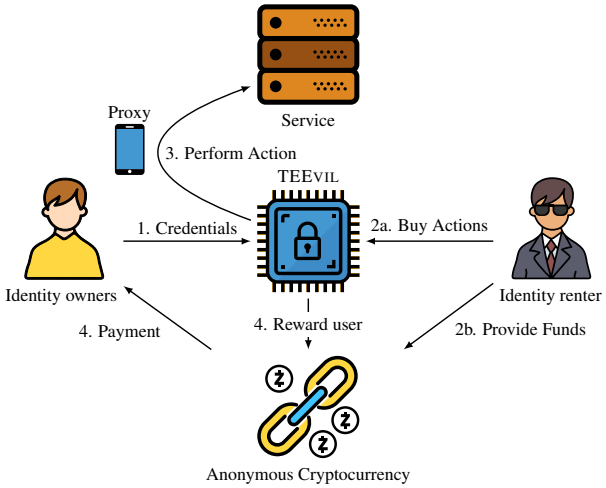


Figure 1: Overview of the TEEVIL protocol showing how its entities interact with each other.

The TEE provides three important properties to TEEVIL: (i) it protects the details (e.g., credentials) of the parties; (ii) it allows to automate all actions that the worker should perform (that are now performed by the verifiable code), and removes the need for manually proving and verifying the execution of actions; finally (iii) it provides fair exchange by managing funds with a decentralized anonymous cryptocurrency such as ZCash. As it provides automatic execution of actions by the TEE from leased accounts, TEEVIL greatly extends crowdturfing by offering better properties, and enabling a large-scale marketplace.

In the following, we give an overview of TEEVIL, instantiating the TEE on top of which it is built with Intel SGX and using ZCash as the anonymous cryptocurrency.

3.1 TEEVIL Overview

In Figure 1 we show the main interactions within TEEVIL. Identity owners enroll in the system in step 1, by securely sending their credentials of the target service and their public cryptocurrency address to the TEEVIL SGX enclave. In step 2a, identity renters start a new campaign by communicating to the enclave which actions they wish to buy from identity owners, and in step 2b, by transferring funds to an address controlled by the enclave on an anonymous public blockchain. In step 3, the enclave automatically performs actions by using the identity owners' devices as proxies, so that from the service's point of view the action appears as any other normal activity of the identity owner. Step 3 is repeated for every user enrolled in TEEVIL. Finally, in step 4, as soon as the enclave receives the confirmation that the action performed in step 3 was done successfully, it issues a transaction to the blockchain that transfers the reward to the relevant identity owner.

3.2 TEEVIL System Details

We now present the protocols and message flow within TEEVIL. To facilitate the analysis, we describe the protocol as if a single infrastructure maintainer is present. We discuss in Section 7.1 how this design can be distributed to support multiple infrastructure maintainers.

The TEEVIL protocol consists of three main parts, depicted in the right-hand side of Figure 2: enrollment of identity owners, campaign creation, and automatic interactions. We outline each one of them in the following sections.

3.2.1 Enrollment of Identity Owners

Three prerequisites have to be met by a user to enroll in TEEVIL as an identity owner: the user needs to have credentials for a valid account for a service supported by the TEEVIL enclave, a cryptocurrency address to receive rewards, and a proxy on one of its devices.

The TEEVIL enclave exposes a web interface in which the identity owners can create a TEEVIL account, where all the user's information can be entered. The connection is secured with a TLS connection whose endpoints are the identity owner's device and the enclave, such that not even the OS in which the enclave is running can observe the data exchanged by the identity owner. The code of the enclave is public, and as part of the connection establishment, the identity owner checks the attestation of the enclave to make sure that it is communicating with a legitimate TEEVIL enclave running inside a genuine SGX CPU.

Once the identity owner has established a secure connection with the TEEVIL enclave, it provides the credentials of its proxy server (if any are specified). Before proceeding further, the TEEVIL enclave ensures that the proxy is working correctly by sending a nonce to itself trough the user's proxy. As soon as the proxy is operational, the user can start providing the credentials and policies of the various accounts it wishes to rent out.

The enclave provides a list of services to the identity owner, for example various social media platforms. For each service, the identity owner inputs his own credentials, and selects a set of policies. The policies are specific to each service, and can restrict the type of content that can be associated with the identity owners through TEEVIL, or what kind of actions will be performed on their behalf. For instance, policies can be set to allow only "likes", or only pictures to be posted on a social network, or, in an e-voting platform, vote for candidates only if they belong to a particular political party. The enclave then checks the credentials of the service by trying to log-in to each one of them. After a successful login the service is considered as successfully enrolled.

The identity owner finally provides the cryptocurrency address to receive rewards. From this point on the various accounts entered by the identity owner can be rented by identity renters.

3.2.2 Campaign Creation

The TEEVIL enclave exposes a separate web interface to identity renters. The connection to this web interface is set up in the same way as described in Section 3.2.1 for the identity owners. That is, the TLS endpoints are at the identity renters browsers and in the enclave, and before exchanging any secret the identity renter attests to the code of the TEEVIL enclave.

As opposed to identity owners (cf. Section 3.2.1), identity renters do not to create any account with the TEEVIL enclave and do not need to have any proxy installed. Identity renters are provided with a set of supported automatic actions for each service. As an example, for a social media campaign, they can provide a link to a post and the number of likes they wish to reach on that post, or they can provide a user and the number of new "followers" they want that user to obtain; in an e-voting service, identity renters can provide the name of a candidate and the number of votes they want it to receive.

Five main steps have to be completed to start a campaign. First, the identity renter fills in the campaign details. Second, the enclave displays the expected price[3] for the specified campaign. Third, the identity renter covers the amount of the campaign and an upfront deposit by transferring money to the cryptocurrency address of the TEEVIL enclave. The deposit will be refunded to the identity renter during the campaign, and we use it to guarantee fairness, as explained in Section 4. Fourth, the identity renter provides its cryptocurrency address, a transaction ID, and the latest block of the blockchain to the enclave. Finally, once the TEEVIL enclave verifies that the transaction exists, it has enough confirmation blocks and that the amount transferred is sufficient, it starts the campaign.

3.2.3 Automatic Interactions

As soon as the campaign begins the enclave starts contacting identity owners whose policies are compatible with the actions of the campaign, we call the accounts of the identity owners that are compatible with the campaign "*compliant accounts*". For each identity owner that has a compliant account, the enclave performs two main operations. First, it uses the proxy in the identity owner's enrolled device to get the latest block from the point of view of that identity owner. If the retrieved block is *not* consistent (cf. Appendix A.2) with the one sent by the identity renter at campaign creation, the TEEVIL enclave does not perform any action for this identity owner in this campaign. Second, if the previous step was successful, it connects to the service through the identity owner's proxy. By doing so, from the point of view of the service the connection appears as if it is initiated from the identity owner's device. Once the TEEVIL enclave establishes the connection with the service, it performs the action requested in the campaign on behalf of the identity owner. The enclave waits for a con-

[3]The enclave always provides an upper bound, since the actual price is determined by the individual price of each account used.

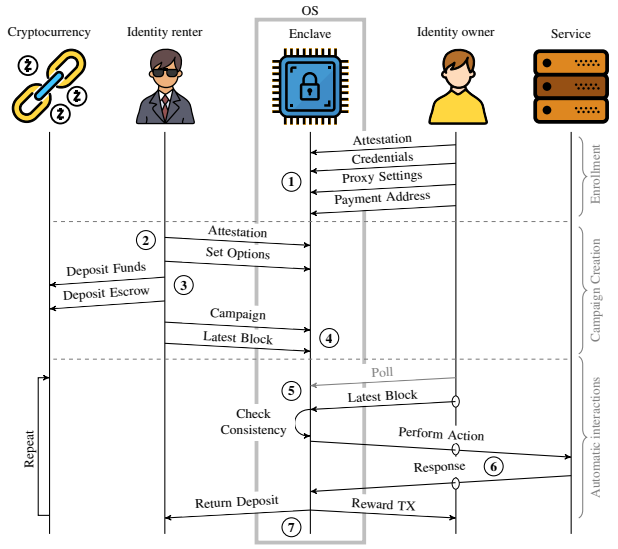


Figure 2: The protocol of TEEVIL.

firmation from the service. For some types of actions, the identity renters might require that the action is not manually reverted by the identity owners within a specified time-frame, in these cases the enclave delays the check until the end of the defined time-frame. Upon successful verification, the enclave issues a transaction on the blockchain that pays the identity owner in the address specified during enrollment and returns a share of the deposit to the identity renter.

When the target number of actions of the campaign is reached, or if there are no more compliant accounts to employ, the enclave terminates the campaign. As part of the termination, any remaining campaign funds and the rest of the deposit are returned to the identity renter on the cryptocurrency address specified at campaign creation.

3.3 Protocol

We now put together all the steps of the protocol described in the previous subsections. We depict the general flow of a TEEVIL campaign in Figure 2, where each number in the figure corresponds to the following steps:

1. Identity owners enroll in TEEVIL. They attest the enclave and provide the enclave access to a proxy running on their devices, their credentials for various services, and a set of policies for these services.

2. An identity renter connects to the enclave and specifies the details of a campaign.

3. The identity renter provides the funds necessary for the campaign and a deposit to the TEEVIL enclave's cryptocurrency address.

4. The identity renter sends the final campaign confirmation and the latest block to the enclave. The enclave then checks if the funds are received correctly and selects compliant accounts to fulfill the campaign.

5. The enclave fetches the latest block through each compliant accounts proxy and checks the consistency with the block provided by the identity renter.

6. The enclave performs the action via the proxy on the identity owner's device.

7. The enclave issues the reward transaction to the respective compliant account. A share of the deposit is immediately returned to the identity renter. If specified by the identity renter, the enclave waits a specified time and checks that the action has not been reverted before issuing the transaction.

Once the campaign is over, or if no more compliant accounts can be reached, the enclave returns the remaining campaign funds and deposit to the identity renter.

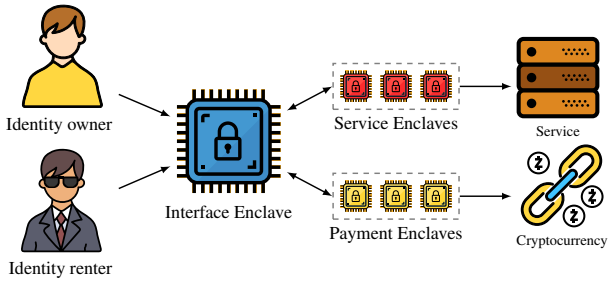


Figure 3: Architecture of the TEEVIL enclaves.

3.4 The TEEVIL Enclaves

So far we considered the TEEVIL enclave as a single enclave that takes care of all the interactions between all the entities in TEEVIL. However, in the interest of modularity (and later scalability), we split the enclave into three parts: (1) the interface enclave, (2) the service enclave, and (3) the payment enclave. We depict them and how they interact with each other in Figure 3. Note that there is an enlistment phase in which the enclaves attest each other, and at this time they exchange their public keys to later establish a secure TLS channel between them without having to repeat attestation.

Interface Enclave. The interface enclave is the entry point to all the other enclaves: Users of TEEVIL (identity owners and identity renters) will use it to start campaigns and enroll their credentials. The interface enclave stores all enrolled credentials and synchronizes the actions of the service enclaves and the payment enclaves. During an enlistment phase the interface enclaves allows an administrator to enroll service and payment enclaves.

Service Enclave. A different service enclave exists for each supported service. For instance, to support a particular social network, a service enclave that knows how to interface with that social network needs to be developed. The service enclave exposes a list of *service specific* tasks and policies to the interface enclave. The attestation report of all service enclaves is also exposed by the interface enclave, so that users can decide to trust only a particular implementation of a service enclave.

The service enclave receives proxy configuration parameters, credentials, and policies from the interface enclave and tries to fulfill the received tasks by contacting the service through the identity owner's proxy. It then sends confirmations of completed actions back to the interface enclave. The service enclaves do not keep any permanent information about the identity owners data. Each service enclave is independent of each other, and allows to scale the number of requests made to the service, as the more enclaves there are the more tasks can be performed in parallel.

Payment Enclave. Similar to the service enclaves there can be multiple implementations of payment enclaves, in this case, each supporting a different digital currency (for more details see Appendix A). However, to ease explanation, we assume there exists a single implementation that supports ZCash.

At the beginning of the campaign, funds are split into smaller shares each of which is controlled by one payment enclave which can then issue transactions independently. To guarantee that all the reward transactions are correctly issued, all transactions of one payment enclave depend on each other: each following transaction uses the unspent output of the previous transaction. Since a share of the deposit is released with every reward transaction, we can guarantee that if the entire deposit is returned then all rewards have been paid as well.

The payment enclaves share a backup copy of their private keys to access the funds with the interface enclave. In the case of a crash of a payment enclave, the interface enclave can use said keys to restore the funds.

4 Security Analysis

In this section we informally analyze the security of TEEVIL through provided guarantees of the three main system properties – fairness, indistiguishability, and plausible deniability (cf. Section 2). We look at two different adversary models. First, we consider the case in which every protocol participant and third parties try to violate fairness. Second, for the latter two properties, we consider a powerful attacker that wants to expose the participants. Thus, the various protocol participants can cooperate to prevent disruption of TEEVIL.

Regarding TEEVIL enclaves, we assume the standard SGX adversary model: the adversary controls the OS on the platform where the enclaves run, and can tap into the memory bus, but cannot tamper with the CPU package. Additionally, the adversary can delay or drop any network packet – however, she cannot see or modify the content of packets if the enclave serves as the TLS endpoint. We assume no adversary can compromise the enclaves and we trust the TEE manufacturer (i.e., Intel) to not fake attestation queries, or otherwise compromise the security of any SGX capable CPUs, either voluntarily or under coercion from an external party. Rollback attacks are out of scope – we refer the interested reader to [12, 58].

4.1 Violating Fairness

Parties participating in the protocol are interested in violating fairness to get economical advantages. However, external entities (e.g., services targeted by TEEVIL) can also be interested in violating fairness in the hopes of damaging the reputation of TEEVIL and push identity owners and identity renters to leave the platform. We first detail the considered threat model, then analyze security against protocol participants, and finally against external adversaries.

Threat Model. All three protocol parties want to break fairness, i.e., the identity renter wants to obtain actions without paying for them, the identity owner wants to get rewards without performing actions, and the infrastructure maintainer wants to get fees without doing the required work. Parties might also collude to break fairness. We assume that protocol parties act rationally and are moved by economic reasons, thus would not do any action that could impact their stakes in the system. However, there might be external adversaries compromising any of the parties – such entities can act without the fear of economic repercussions and can make the compromised parties act against their interest. We do not provide any fairness guarantee to these compromised parties, and analyze how they would impact the remaining honest parties.

Given adversarial capabilities and the assumed SGX adversary model, we highlight that no misbehaving party can impact the protocol by modifying execution of the TEEVIL enclave at any step, or by modifying the content of network messages. Hence, an adversary can only affect fairness by cutting network messages of the protocol: Figure 4 depicts all susceptible messages.

Protocol Participants. We now analyze the fairness guarantees of TEEVIL when only protocol participants try to cut connections and messages. The identity renter initiates a campaign by sending its current view of the blockchain to the enclave. The identity renter has full control over the content of this message, and could send the TEEVIL enclave a different view of the blockchain if he so wishes (message (1) in Figure 4). Then, by cutting message (2) of Figure 4, the identity renter could try to trick the enclave into performing an action without actually having the funds. The attacker hence effectively forges the enclave's view of the blockchain. However, TEEVIL will only perform an action on behalf of

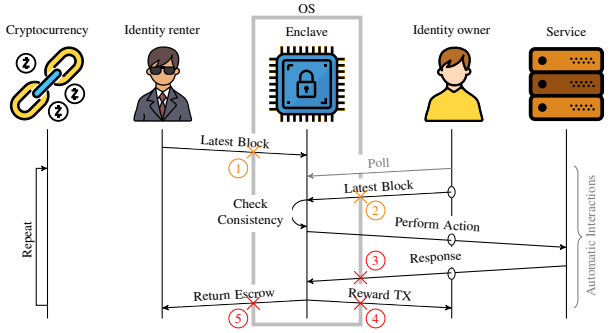


Figure 4: Adversaries controlling the OS of the infrastructure maintainer, or the local network of protocol participants, can cut network connections at the points marked 1–5 to try to disrupt the protocol and its properties. Note that the content of message 1 and 2 can be modified by the identity renters and identity owners, respectively, since there is no TLS connection to the blockchain nodes.

an identity owner if and only if the blockchain state from the identity owner is consistent with the one provided by the identity renter. If an adversary can carry out a successful double spend attack against some identity owners, then he can break the fairness property for these specific parties, but not against other unaffected identity owners.

The identity renter might also try to cut the response of the external service, that is message (3) in Figure 4. The enclave will then never get a confirmation of the completed action but it will also not get an error message, and it is, therefore, able to distinguish an attack from incorrect credentials[4]. In this case, the identity renter would lose a part of his deposit. Messages (4) and (5) in Figure 4 are intrinsically linked since both the reward and the share of the deposit are linked into a single transaction. By blocking this transaction, the identity renter would lose his share of the deposit.

Any of the previously mentioned messages could also be blocked by the infrastructure maintainer. However, he would forfeit his fees, since the infrastructure maintainer gets the fees with the reward transaction (messages (4) and (5) in Figure 4), so he has incentives to keep the system running as long as possible.

In conclusion, tampering with any of the five messages highlighted in Figure 4 would result in losses for all parties (even when colluding): identity renters (escrow), identity owners (reward) and infrastructure maintainers (service fees) – thus, no rational party would perform this attack.

External Adversaries. As discussed, an external adversary can compromise any protocol party and make them act irrationally to try to break fairness for other parties (no guarantee of fairness is given to the compromised party). We now analyze what happens to the remaining protocol participants:

- A compromised **identity renter** cannot impact the fairness for any other party, since it can only tamper with messages (1) and (4) of Figure 4. As discussed, tampering with message (1) has no implication for identity owners that have the correct view of the blockchain, and cutting message (4) does not prevent the reward to the identity owner because she receives a copy of this transaction as well in message (5).
- A compromised **identity owner** might cut message (3) and forfeit the reward in order to destroy a small share of the deposit, thus hurting the identity renter.
- A compromised **infrastructure maintainer** could either shut down the enclave upon receiving the funds, or he could cut messages (3) or the reward transaction in Figure 4, thus executing all actions but not paying any reward and destroying the deposit.

Additionally, a service targeted by TEEVIL is in the unique position of being able to fake the proof of performed action (message 3 of Figure 4). This can happen if the service colludes with identity owners (for example by registering a large set of fake identity owners to a TEEVIL enclave), therefore it knows when it receives a rented action and provides fake responses accordingly. If the result of the rented action is observable, such as a "like" on a OSN, this is trivially prevented: the enclave can verify "externally" (e.g., through a legit account) if the action was performed. However, if the result of the action is not observable, like in e-voting, the service can violate fairness successfully. We discuss a Peer-to-Peer design for TEEVIL that can mitigate this scenario by requiring one SGX CPU per enrolled account in Section 7.1.

4.2 Exposing Parties

Services targeted by TEEVIL might try to limit its impact by breaking *indistinguishability* and filtering fraudulent actions. In the case of e-voting, the service provider (i.e., a government) might also try to punish participants of TEEVIL with legal repercussions by breaking *plausible deniability*. Here we first elaborate on such a threat model, then analyze the security against non colluding and colluding adversaries.

Threat Model. Indistinguishability and plausible deniability require a different threat model: uncompromised protocol participants might want to cheat on each other regarding fairness, but ultimately they still want to perform the exchange of money for identities. As they have monetary stakes in the system, they will not try to expose each other without any external influence, and can instead cooperate to avoid being exposed. Therefore, the adversaries we consider here are external entities that can possibly compromise protocol

[4]Note that identity owners could try to fake message (3) by exposing a controlled copy of the target service to the enclave – however, this can be prevented by existing countermeasures such as certificate pinning [32].

participants (to expose other participants, as the exposure of the compromised party is trivial), and the services targeted by TEEVIL. All these adversaries can collude to reach their goal.

Once again, we assume the previously described standard SGX adversary model where the adversary can cut connections and power down enclaves, but the enclaves execution or the messages cannot be modified. Additionally, services have full control over their platform, and can run any kind of analysis on users' actions to try to find accounts that have been rented. As a discussion point, we also consider *global* network attackers with a complete view of the network.

No Collusion. When assuming no collusion, the adversary is in full control of a single entity (e.g., the service) but cannot cooperate with any other entity. For *indistinguishability*, assuming that protocol participants do not try to expose each other, the only possible adversary is the service. Indeed, services targeted by TEEVIL are the only ones that can analyze users' actions – and thus can try to distinguish between fraudulent (i.e., rented) and genuine actions. To do so, services can analyze both users' behavior on their platforms, to understand if actions are performed automatically or by a human; services can also analyze the result of users' actions, e.g., posted or "liked" content, to try to detect fraudulent actions. These are active research areas, proposing techniques such as CAPTCHAs [82], or Machine Learning tools to build behavioral and content profiles to detect misbehaving users [40, 80] and their actions [47, 63]. The literature on these topics is extensive – we report the main research results in Section 5.1, where we discuss that there is still no conclusive countermeasure to stop systems such as TEEVIL. Here, we only highlight that such areas of detection are subject to cat-and-mouse games, where adversaries can often avoid detection by changing their strategy after defenders deploy their countermeasures [50, 80, 84, 90].

Colluding Adversary. A stronger adversary can collude with any party and, is possibly, a global network adversary. The service can collude with identity renters, and create campaigns with specific targets that therefore de-anonymize identity owners (e.g., a OSN asking to "like" a post that is only visible via a direct link, to prevent legitimate interactions). Another possible attack includes observing all the connections to and from any TEEVIL enclave, thus correlating them to the identity owner's proxy and to the service. Given the ability to then link these connections with the service's accounts it would be possible to identify identity owners participating in TEEVIL, breaking both indistinguishability and plausible deniability. Even adding an anonymity network (e.g., Tor, as we discuss in Section 7.2) does not mitigate against this attacker, since a global network attacker is outside the Tor threat model. Therefore under this attacker model, for services that allow to link actions with identity owners (e.g., social networks, but *not* e-voting), TEEVIL cannot guarantee indistinguishability and plausible deniability.

4.3 Anonymity

In the general case, all operation on TEEVIL are anonymous: identity renters do not learn any detail about identity owners, whose credentials are securely stored and not accessible even by the infrastructure maintainers. However, there could be specific actions and target services that can violate anonymity. For example, when the bought action has an observable result (e.g., interactions on OSNs, as opposed to e-voting), identity owners could see who is the target, and identity renters could try to correlate the start of campaigns and the results of the bought actions. Note that, even in this case, both parties can still appeal to plausible deniability. More powerful adversaries are infrastructure maintainers, who could track IP addresses of identity owners and identity renters; in general, powerful network adversaries could correlate IP addresses connecting to the enclave and subsequently to target services. If such a strong adversary would collude with the target service, they could directly expose users' rented accounts, and act accordingly. Network adversaries can be mitigated (with the exception of timing correlations) by using mix networks such as Tor [28], as we discuss in Section 7.2. While there is no general solution to deanonymization by specific actions, identity owners can specify policies regarding the actions that they are willing to sell, thus specifying the amount of risk they are willing to take.

5 Defenses

We now shift perspective and present some possible defenses against systems such as TEEVIL. These countermeasures extend our reasoning presented in Section 4, and follow from the most powerful adversary models we considered. We discuss existing countermeasures for the two applications we considered throughout this paper (OSNs and e-voting), and then elaborate on potential future measures. We attempt to order defenses by increased complexity for the defending entities, and highlight that all countermeasures require significant effort for the defenders. In particular, some countermeasures require an unrealistic amount of collaboration between nations and services while others only lessen the impact but do not entirely prevent the problem. In conclusion, we believe that none of these countermeasures is easily employable to defend against systems such as TEEVIL, and more research attention is necessary to stop this threat.

5.1 Existing Techniques - OSNs

There already exist countermeasures against "non-human" accounts and interactions on OSNs, both deployed by commercial providers and proposed in the literature. We coarsely group them according to what OSNs can analyze: (i) users'

behavior, trying to understand if the actions were performed automatically, and (ii) users' interactions, e.g., posted content, or targets of "like" actions, trying to uncover automatically generated content by TEEVIL.

Users' Behavior. The behaviour of authentic users can be fully imitated by TEEVIL. Detecting rented accounts is thus identical to the conventional bot detection problem. We discuss two aspects of this problem and the relevant literature next. First, services might try to use CAPTCHAs [82] to detect automated actions. However, CAPTCHAs can be bypassed by advances in machine learning technologies [90], or by offloading them to cheap labor [62]. Services might also build users' behavioral profiles and try to detect "compromised" accounts; however, most approaches presented in literature are based on machine learning [3] – turning into a cat and mouse game where detection can be easily avoided by changing strategies [80, 84]. Second, services might try to detect if users' behavior changes abruptly, or deviates for a long time from previous behavioral profilings [3, 40]. However, in a platform such as TEEVIL, rented actions are interleaved with identity owners' benign behavior, and it is thus unclear if any of these solutions applies to identity lease. OSNs can instead try to detect behavior similar to TEEVIL's identity owners: *brief* deviations from normal behavior [80], and then try to correlate such deviations among multiple accounts [31] in order to expose vast campaigns. Proposals in the area still rely on hand-crafted features that TEEVIL can manipulate, and plausible deniability still applies: identity owners can claim it was their legitimate interaction, therefore it is hard to argue that OSNs can easily start deleting posts and interactions. Ultimately, we note that even though fake and compromised accounts are a well studied problem in research, OSNs still struggle to detect them [34, 77], showing that this problem is far from trivial.

Users' Interactions. Services are interested in trying to detect crowdturfed and fake content. There has been a lot of effort in detecting such content [47, 66], accounts involved in crowdturfing activities [63], and even the targets of crowdturfing actions [74]. However, this is again a cat-and-mouse game, as recent efforts have shown how to evade such classifiers [84] and how to automatically generate realistic content that can fool even human readers [50, 89]. To conclude, some of the arguments for (i) still apply: plausible deniability is not violated, as there is no proof that identity owners did not generate the dubious content, and these countermeasures are even ineffective if there is no observable result.

5.2 Existing Techniques - E-voting

In this section we analyze how some security properties of e-voting schemes might defend against TEEVIL by preventing its operations, or by violating TEEVIL's requirements. In particular, here we look at three properties of e-voting and we refer the interested reader to Appendix A.4 for more details. First, verifiability, and in particular *individual verifiability*, gives the voters confidence that their vote was recorded as intended. Second, coercion-resistance removes the possibility from voters to prove to third parties how (and whether) they voted. Finally, *privacy* is a weaker form of coercion-resistance which prevents votes from being linked to a specific voter.

We premise our analysis by observing that, regarding interactions between an e-voting service and TEEVIL, the TEEVIL enclave acts as both the device with which the user votes, and as the voter itself, from the point of view of the voting service. This behaviour is referred to as a *simulation attack* [49] in the literature since the enclave can perfectly emulate a voter. We note that voters successfully enroll as identity owners by providing all the secret that they needed to register and participate in the election.

Privacy. This property inadvertently helps the indistinguishability and plausible deniability properties of TEEVIL. Because of voter privacy, it is very challenging for the voting authorities to distinguish between voters giving their inputs to a legitimate voting device, and the TEEVIL enclave autonomously voting on behalf of a voter – because the TEEVIL enclave can be seen as a voting device. Another consequence of voter privacy is that even in the extreme case in which the election authorities could tell which votes were cast with TEEVIL[5], they would not be able to link them to specific voters.

Verifiability and Coercion-Resistance. Verifiability [22, 54] and coercion-resistance [19, 20, 49] have an impact on the fairness property of TEEVIL. In particular, the former aids it, while the latter could break it, thus potentially being a concrete defence for e-voting against TEEVIL.

A distinct difference from e-voting to OSNs is that in e-voting a identity owner can only sell one single action per election (i.e., one vote), as opposed to OSNs where usually actions can be performed and reverted an arbitrary number of times. Individual verifiability, in this case, helps to achieve fairness: since the TEEVIL enclave acts as the voter, it can check whether a vote has been cast and whether it reflects the choice that the identity renter is willing to pay for. This aspect is particularly relevant when one considers the different policy with regards to multiple ballots submissions for a voter. In particular, for each voter two policies are common: (i) only the first vote counts and all subsequent votes are discarded, (ii) only the last submitted ballot counts, and all previous ones are ignored. Individual verifiability allows to check whether any vote was submitted before (or whether no vote was submitted after, respectively) the TEEVIL enclave casts its vote.

Coercion-resistance [19, 49] is the most promising property to defend against TEEVIL. However, while coercion-resistance makes it impossible to prove to a third party that

[5]For instance, because the election allows write-ins that were used only when voting with the TEEVIL enclave.

a voter voted in a particular way, the voter itself (in order to guarantee individual verifiability) has to be able to check the content of its own cast ballot. Therefore, *if the* TEEVIL *enclave has valid voting credentials*, since it acts as a voter, coercion-resistance does little to prevent the enclave to verify that the vote was cast correctly - this is effectively a simulation attack. Hence, in order to prevent a simulation attack, in [49] it is proposed to allow the voter to generate arbitrary *invalid* voting credentials which are indistinguishable from the real ones (the votes cast with these credentials are later discarded). However, since the initial credentials are delivered to the voter by the election registration authorities, the TEEVIL enclave could simply ask for the transcript of this interaction to verify that the credentials are the valid ones[6].

Receipt-freeness is a weaker property than coercion-resistance, which is often looked at in the literature [43, 55, 56, 61]. On the other hand, it does not help against TEEVIL, since it requires the voter to vote on its own and without being observed by the coercer (usually by assuming an untappable channel to the voting servers). This is because TEEVIL votes on behalf of the voter. Therefore, since TEEVIL violates one of the key assumptions of receipt-freeness, any protocol providing this property cannot impact fairness in TEEVIL.

In conclusion, most properties of e-voting schemes inadvertently make TEEVIL more robust, thus making the design of effective defenses very challenging for this application. Any protocol providing privacy and individual verifiability - two key properties of any e-voting protocol - strengthens the fairness, indistinguishability, and plausible deniability properties of TEEVIL. Receipt-freeness, a property suggested multiple times to combat vote selling, is completely ineffective against TEEVIL. In many coercion-resistant schemes it is assumed that the coercer (i.e., TEEVIL) cannot simulate the voter during registration [20, 49], and is not always physically collocated with the voter [27]. Only when these two assumptions hold[7], coercion-resistant protocols break the fairness property of TEEVIL.

5.3 New Defenses

Enroll Fake Accounts. Any service could easily create "ghost"accounts: profiles that seem genuine but do not affect anything on the service, e.g., such accounts could post content that no one can see in an OSN. Thus, by enrolling a large number of such accounts, the service can try to lessen the impact and make campaigns less effective. However, TEEVIL could use other accounts to verify the actions. This however, would still be effective for e-voting and other applications in which the identity owners' actions are not observable. However, some decentralized designs of TEEVIL could make this mitigation too expensive to deploy, as we will discuss in Section 7.1.

Provide Incentives for Identity Owners. Identity owners participate in TEEVIL to gain monetary rewards. The service could play the same game and offer compensation for revealing their participation in TEEVIL, undermining the potential impact of rented accounts. However, this is a clear conflict of interest for the service and potentially results in all users wanting to receive some compensation.

Start Campaigns to De-anonymize. The service could start his own campaigns on very specific content which might not even be reachable without a direct link. The users who then perform an action on the prepared resource are directly confirmed to be identity owners in TEEVIL and can be put under increased monitoring. Note that this would not be effective for applications in which the actions are not linked to identity owners (such as e-voting). In any case, the service would have to spend money and pay each identity owners a small amount for revealing themselves. Additionally, TEEVIL could require a set of actions to go to the content instead of a single link.

Compromising Every Identity Owner's OS.

If a government is interested in identifying the identity owners of a system like TEEVIL, because for instance it suspects that they are being used to compromise its election process, it is not difficult to imagine that it would try to convince a OS vendor to monitor all its users to discover which ones are enrolled in TEEVIL. However, monitoring every action of every user's device for the sake of identifying users that might be participating in TEEVIL would pose serious concerns for the privacy of all citizens.

Global Network Attacker. Another way to expose parties involved in TEEVIL is by observing the network and correlating activities of TEEVIL and identity owners, for example by correlating timing. To do so, services and governments would need to be in control of large portions of the network. However, this leads to worrisome consequences on privacy and censorship: governments need to control the Internet over their whole country to expose TEEVIL. Moreover, the reach of nation state adversaries is usually limited to their sovereignty – if TEEVIL would distribute its infrastructure over rival countries, cooperation between these nations is unlikely.

6 Implementation

We implemented a prototype of TEEVIL using widely available Intel SGX enclaves. Namely, we implemented the interface enclave, a service enclave for *Reddit*, and a payment

[6] Juels et al. [49] assume that this transcript can be deleted by the voter, but in TEEVIL the voter wants to sell his vote and can choose to not delete the transcript.

[7] For instance, the registration phase can be done in person, thus preventing the attacker from obtaining a transcript with the initial credentials of the voter. However, the case in which TEEVIL has all the secrets of the voter is equivalent to the case in which it is physically collocated with it since it can check how and whether a voter voted.

enclave for *ZCash*. In our implementation, a identity renter can buy "upvotes" on Reddit posts, for which identity owners are rewarded with a payment in ZCash. The interface enclave and the service enclave are implemented in `C++` using the Intel SGX SDK [45], while the payment enclave is built using `rust`. All enclaves communicate with each other and external parties using TLS. Note that the TLS endpoint lies in the enclaves and the OS is only responsible for the TCP/IP protocol.

Interface Enclave. The interface enclave uses a prototype storage backend similar to a very minimal database to store a large number of identity owner credentials while still keeping the data under the protection mechanisms of Intel SGX. The interface enclave exposes a RESTful API interface to external parties which supports all the previously described interactions with TEEVIL. It also supports multithreading for improved performance.

Service Enclave (Reddit). To successfully perform actions on the identity owner's behalf, it's important for the enclave to behave as close to a genuine user as possible, performing all the necessary POST/GET and cookie exchanges just as the browser would. The service enclave exposes a REST API, just like the interface enclave, and the confidentiality of the parameters (most notably, the identity owner's credentials) exchanged from one enclave to the other is ensured by TLS.

Payment Enclave. It is now feasible to create shielded ZCash transactions in an Intel SGX enclave, as generating a zk-SNARK only needs 85MB of memory in our measurements[8], thanks to the 2018 "Sapling" update [11]. The payment enclave uses a modified version of the *bellman* library for the required zk-SNARKs [92], which is also used in the official *ZCash* implementation. Since Intel SGX enclaves do not support the standard POSIX multithreading API, we run the zk-SNARK generation on a single core.

To create ZCash transactions, the payment enclave needs to get the Merkle paths to the notes[9] that it wants to spend. We use the scheme described in [88] to keep up to date paths to all spendable notes.

Proxy. We use a proxy on a device of the identity owner to tunnel the requests of the service enclave. We note that identity owners do not necessarily need to manually set up and configure a proxy on their devices. The process can be automated, e.g., by providing pre-configured apps, or by downloading a configuration file (which can be made available through the enclave website) on their device – so the only information they need to enter manually is related to the service's credentials and policies.

The devices the identity owner uses for this proxy might be behind a NAT or change IP address. Therefore we require a regular polling message to the interface enclave that updates the proxy information of the respective identity owner. Such polling message could be done manually by connecting again to the enclave web interface, or automatically by letting the identity owners install an app in their devices.

[8] SGX enclaves so far support only up to 128MB of memory. Swapping allows to use more memory but severely hurts performance.

[9] A note in ZCash is similar to an output in Bitcoin.

	Baseline implementation		SGX	
	Average [s]	Std [s]	Average [s]	Std [s]
1st request	0.971	0.295	1.202	0.249
2nd request	0.370	0.158	0.402	0.128
3rd request	0.663	0.175	0.769	0.197
4th request	1.581	0.275	1.560	0.298
5th request	0.280	0.206	0.355	0.329
Total	3.865	0.511	4.288	0.561

Table 1: A step-by-step timing of a Reddit "upvote", divided into the 5 required network requests, performed by our SGX service enclave compared to a baseline implementation outside the enclave (sample size: 100 "upvotes"; ping: 13.2ms).

Scalability. The architecture of TEEVIL is designed with scalability in mind: it supports many service enclaves and payment enclaves that run concurrently. The interface enclave transmits a list of tasks to the service and payment enclaves which then try to execute all of the tasks in an asynchronous manner. After that, they respond with a confirmation for all correctly executed tasks.

6.1 Performance Evaluation

We now evaluate the performance of our implementation of TEEVIL on a desktop with an i7-8700k. For experiments that require a second machine we use a separate machine with an i7-7700K. Since we do not have access to a large amount of credentials for Reddit, we split the evaluation into three parts: Reddit actions, inter-enclave communication, and zk-SNARK generation. We then show how to combine these results to obtain an estimate for the complete system.

Reddit. We repeatedly performed an "upvote" on a controlled selection of posts using four accounts created specifically for this purpose. We present the aggreated results over 100 measurements in Table 1. Note that we show the distinct measurements of the 5 different requests (from logging in to the service, to finding the content and interacting with it) that make up a single "upvote" action.

Every service enclave also needs to check the consistency of the ZCash blockchain state provided by the identity owner and identity renter. Our implementation verifies the proof-of-work and hash chain for 10 blockheaders in 2.9ms on average, which is negligible compared to the total time.

TLS Between Enclaves. We measured the throughput and latency of TLS connections between two enclaves to show

	Time [ms]		
	1 Thread	4 Threads	8 Threads
Normal	4521 ± 43.2	1322 ± 26.6	975 ± 37.4
Intel SGX	4935 ± 114.1	✗	✗

Table 2: Zk-SNARK performance comparison between Intel SGX and normal operation (sample size: 100 proofs).

how the interface enclave would communicate with service and payment enclaves. Our implementation manages to complete 56.3 TLS handshakes per second between two enclaves with 8 threads each using cipher suite `TLS-ECDHE-RSA-WITH-AES-256-GCM-SHA384`. Thus, TEEVIL can scale to a large number of service and payment enclaves.

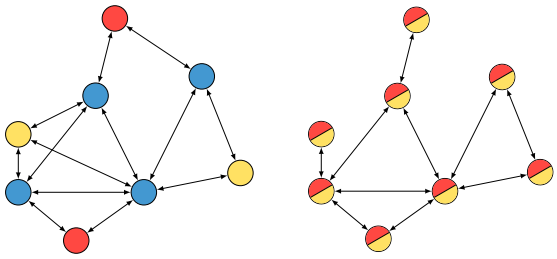
zk-SNARKs. We compare the time spent to generate a zk-SNARK proof in the unmodified *bellman* library and inside Intel SGX in Table 2. Notably, Intel SGX adds about 10% overhead compared to normal operation. In our application multithreading is not necessary since we require a high throughput of zk-SNARKs per second and do not optimize solely on latency. A higher throughput can easily be reached by spawning more payment enclaves.

Example. To show how to combine our performance evaluation into an accurate estimate for a specific instantiation of TEEVIL, we present an example in the following.

Let's assume an identity renter wants to start a campaign on Reddit with 1000 identity owners. TEEVIL is instantiated with one interface enclave, 25 service enclaves, and 25 payment enclaves. The interface enclave first splits the 1000 actions into batches of 40 actions each, and then sends one batch to each service enclave. Each service enclave then tries to fulfill all actions, and sends a confirmation report back after about 160s. In these 160s all requested actions should be performed on Reddit, ideally boosting popularity of the content specified by the identity renter. After this period of time, the interface enclave transmits the list of confirmed actions and blockchain addresses of the identity owners to the payment enclaves. The payment enclaves will then issue all the reward transactions in about 200s.

7 Discussion

In this section, we first discuss a way to strengthen the current TEEVIL protocol by moving to a decentralized mode, then we look at the addition of anonymity networks, and finally we discuss compromised TEEs.



(a) Distributed TEEVIL. (b) P2P TEEVIL.

Figure 5: Decentralized TEEVIL. Blue nodes are interface enclaves, red nodes service enclaves, and yellow nodes payment enclaves.

7.1 Decentralizing TEEVIL

If any system like TEEVIL were ever to be realized, compromising its availability would be one of the first objectives some attackers (e.g., services and governments) would be after. In the current attacker model (cf. Section 4) a single adversary controlling the OS of the infrastructure maintainer can shut-down TEEVIL, by blocking the enclave execution or cutting off all the network connections to the enclaves. Therefore, TEEVIL has a central point of failure: the infrastructure maintainer. In the interest of availability, we sketch two ways in which a fully distributed TEEVIL system could be built, which we call *Distributed* TEEVIL and *P2P* TEEVIL.

Distributed TEEVIL. It is easy to extend the TEEVIL protocol to allow multiple independent parties to run a service enclave and a payment enclave (cf. Section 3.4). To incentivise people to help TEEVIL scale, and be more tolerant to DoS attacks, anyone running service or payment enclaves can get a reward for the actions going through their machine.

Distributing the interface enclave requires a bit more consideration, as they may need to synchronize between each other (e.g., with a gossip protocol). We envision a system in which multiple interface enclaves manage campaigns independently of each other, but keep a global list of identity owners. With this architecture, each enclave type (interface, payment, and service) is connected to at least another interface enclave, and interface enclaves need to have at least one payment enclave and one service enclave connected to them to be operational. We provide an example of this topology in Figure 5a. A more detailed architecture is in Appendix B

P2P TEEVIL. To decentralize TEEVIL in the P2P design, we eliminate the interface enclave altogether. We envision a system in which many identity owners have a machine with an SGX enabled CPU, or equivalent technology that allows to run a Trusted Execution Environment (TEE).

In this design, each identity owner hosts its credentials in an enclave running in its machine. Such an enclave is a combination of the payment and service enclaves: it takes care of the connection to the service, and of rewarding the identity

owner through the blockchain. Note that, in this design, we do not need a proxy since the enclaves will connect to the service directly from the identity owners' device. These enclaves can build a network between them so that a identity renter only needs to contact one of them to start a campaign. We depict the topology of this P2P TEEVIL in Figure 5b.

Upon providing the funds to any one of the TEEVIL enclaves, the contacted enclave will take care of broadcasting the campaign request on the network. Other identity owners' enclaves will then take care of performing the campaign actions if they have the credentials of any compliant account (cf. Section 3.2.3). The TEEVIL enclave is not restricted to serve a single identity owners credential, but it can serve as a node for other identity owners who cannot or do not wish to run a TEEVIL enclave in their machine. These users can delegate their credential to a identity owner which they trust.

We highlight an additional security benefit given by P2P TEEVIL. One of the possible countermeasures to TEEVIL is enrolling a large number of fake identity owners by targeted services. We discussed that, when the result of bought actions is not observable, this violates the fairness property of the protocol. In P2P TEEVIL we could restrict registration of only a single identity owner per CPU, thus requiring the service to invest in CPUs in order to flood TEEVIL with fake accounts. Such solution could be enforced for example by using Intel's linkable attestation protocol [48]. This mitigation would raise the bar for services interested in damaging TEEVIL, but it would also limit its reach, since real identity owners without an SGX CPU will not be able to participate in TEEVIL anymore.

7.2 Adding anonymity networks to TEEVIL

We discussed in Section 4.3 that TEEVIL cannot provide any meaningful guarantees in terms of anonymity. This is based on the fact that *depending on the service type*, the service provider can invest in campaigns that require identity owners to perform some actions that would not be otherwise possible by an identity owner *not* participating in the campaign. Since the service can observe the results of the campaign on a per-user basis, it can then de-anonymize these identity owners, by monitoring which ones performed the action.

This de-anonymization requires the service to be able to monitor the actions of its users, which is in general not possible for every service. E-voting is an instance of such a service type, since it needs to guarantee the anonymity of the vote, and therefore by design it should not be possible to look at the actions of identity owners to correlate them with a custom campaign action.

However, even with services in which identity owners actions cannot be observed, if the service provider can correlate connections from the infrastructure maintainer to the service it might be able to de-anonymize the user. To perform this correlation the service provider either needs to collude with the infrastructure maintainer or to compromise its OS. Observe that in a decentralized version of TEEVIL the attacker could simply run a couple of service enclaves in order to observe connections going to the service from the TEEVIL enclaves.

An anonymity network such as Tor [28] provides two main advantages to TEEVIL. First, it provide more resilience against the attacker introduced in this section, since correlating connections through Tor is only possible for a global network attacker, and compromising two nodes (the service enclave and the service itself) is not enough anymore to track a connection.

Second, if any of the TEEVIL enclaves are running as hidden tor services [28], it would be difficult to localize them and consequently shutting them down, thus providing stronger availability if combined with a decentralized version of TEEVIL. Note that this would give stronger confidence to the users of TEEVIL as well, since they would know that even if the OS of the TEEVIL enclaves is compromised, that enclave would not be able to observe the real IP addresses of both identity owners and identity renters connecting to it.

7.3 Compromised TEEs

TEEVIL relies on the security of the Trusted Execution Environment (i.e., Intel SGX) to protect the confidentiality of its users and guarantee fairness among the different entities which participate in the protocol. Physically compromising a single enclave, or discovering a vulnerability that allows to compromise its remote attestation mechanisms, would jeopardize all the security properties of TEEVIL.

Microarchitectural attacks have surfaced that compromise TEEs [79] and side-channel leakage might allow attackers to infer secret data inside of Trusted Execution Environments [13, 57]. Other attacks, such as physical attacks, are expensive, usually cannot be amortized on multiple processors, and can only be done on devices in the physical possession of the attacker. This implies that in the centralized version of TEEVIL (cf. Section 3) such types of attacks could be carried out only if the attacker is the infrastructure maintainer or if the attacker can gain physical access to the facilities of the infrastructure maintainer. Thus making the attack not only expensive, but also very difficult to be carried out in practice on an already deployed system.

However, these limitations vanish in the decentralized version of TEEVIL introduced above. The attacker can run its own TEEVIL enclave and join the network. Thus it would have physical control over one of the deployed TEEVIL enclaves. In distributed TEEVIL (cf. Section 7.1), physically compromising any enclave leaks every identity owner ever enrolled. While on P2P TEEVIL compromising an enclave leaks all the identity owners that decided to delegate their credentials to the enclave hosted by the attacker. Thus the two version of TEEVIL offer different guarantees in this scenario, and might justify the cost of physically compromising

a processor, depending on how valuable is the information protected by the TEEVIL enclave for the attacker.

8 Conclusions

In this paper we investigated a new type of user monetization through *identity lease* and discussed its potential effects on digital societies. We showed through the examples of OSNs and e-voting that such a system, which combines Trusted Execution Environments (TEEs) and anonymous cryptocurrencies, could allow seamless facilitation of leasing identities via account rental, and subsequently impact the real world. We designed a protocol that, thanks to advances in these technologies, can implement such a system called TEEVIL, and conducted real world tests using legitimate credentials on the Reddit online social network.

We showed how TEEVIL allows, for the first time, creation of a large-scale marketplace which guarantees fairness, indistinguishability and plausible deniability to all the participating parties with acceptable performance.

Such a marketplace could be used to polarize people's opinion by influencing them on online social networks and even to directly compromise e-voting, to name two of its possible applications. Because of the impact that TEEVIL could have on people's online presence and democratic discourse we discussed several defences that could be deployed against systems such as TEEVIL, and note that further research is necessary in this direction.

References

[1] Keystone: Open-source Secure Hardware Enclave, 2018. https://keystone-enclave.org.

[2] Scytl online voting solution, 2019. https://www.scytl.com/en/customers/.

[3] K. S. Adewole, N. B. Anuar, A. Kamsin, K. D. Varathan, and S. A. Razak. Malicious accounts: dark of the social networks. *Journal of Network and Computer Applications*, 79:41–67, 2017.

[4] T. Alves and D. Felton. TrustZone: Integrated Hardware and Software Security-Enabling Trusted Computing in Embedded Systems, 2004. http://infocenter.arm.com/help/topic/com.arm.doc.prd29-genc-009492c/PRD29-GENC-009492C_trustzone_security_whitepaper.pdf.

[5] I. Anati, S. Gueron, S. Johnson, and V. Scarlata. Innovative technology for cpu based attestation and sealing. In *Proceedings of the 2nd International Workshop on Hardware and Architectural Support for Security and Privacy*. ACM, 2013.

[6] E. Androulaki, G. O. Karame, M. Roeschlin, T. Scherer, and S. Capkun. Evaluating user privacy in bitcoin. In *International Conference on Financial Cryptography and Data Security*, pages 34–51. Springer, 2013.

[7] N. Barbieri, F. Bonchi, and G. Manco. Topic-Aware Social Influence Propagation Models. In *2012 IEEE 12th International Conference on Data Mining*. IEEE, 2012.

[8] E. Ben-Sasson, A. Chiesa, E. Tromer, and M. Virza. Succinct non-interactive zero knowledge for a von neumann architecture. In *Proceedings of the 23rd USENIX Security Symposium (USENIX Security)*, pages 781–796, 2014.

[9] A. Bessi, F. Petroni, M. Del Vicario, F. Zollo, A. Anagnostopoulos, A. Scala, G. Caldarelli, and W. Quattrociocchi. Viral misinformation: The role of homophily and polarization. In *Proceedings of the 24th International Conference on World Wide Web (WWW)*, pages 355–356. ACM, 2015.

[10] R. M. Bond, C. J. Fariss, J. J. Jones, A. D. Kramer, C. Marlow, J. E. Settle, and J. H. Fowler. A 61-million-person experiment in social influence and political mobilization. *Nature*, 489(7415):295, 2012.

[11] S. Bowe. Cultivating sapling: Faster zk-snarks–zcash blog. *Zcash Blog*, 2017.

[12] M. Brandenburger, C. Cachin, M. Lorenz, and R. Kapitza. Rollback and forking detection for trusted execution environments using lightweight collective memory. In *Proceedings of the 47th Conference on Dependable Systems and Networks (DSN)*, pages 157–168. IEEE, 2017.

[13] F. Brasser, U. Müller, A. Dmitrienko, K. Kostiainen, S. Capkun, and A.-R. Sadeghi. Software grand exposure: SGX cache attacks are practical. In *11th USENIX Workshop on Offensive Technologies (WOOT 17)*. USENIX Association, 2017.

[14] J. J. Brown and P. H. Reingen. Social ties and word-of-mouth referral behavior. *Journal of Consumer research*, 14(3):350–362, 1987.

[15] V. Buterin et al. A next-generation smart contract and decentralized application platform. *white paper*, 2014.

[16] S. Checkoway and H. Shacham. Iago attacks: Why the system call api is a bad untrusted rpc interface. In *ASPLOS*, volume 13, pages 253–264, 2013.

[17] S. Chen, J. Fan, G. Li, J. Feng, K.-l. Tan, and J. Tang. Online topic-aware influence maximization. *Proceedings of the VLDB Endowment*, 8, 2015.

[18] J. Cheng, L. Adamic, P. A. Dow, J. M. Kleinberg, and J. Leskovec. Can cascades be predicted? In *Proceedings of the 23rd International Conference on World Wide Web (WWW)*, pages 925–936. ACM, 2014.

[19] J. Clark and U. Hengartner. Selections: Internet voting with over-the-shoulder coercion-resistance. In *International Conference on Financial Cryptography and Data Security*, pages 47–61. Springer, 2011.

[20] M. R. Clarkson, S. Chong, and A. C. Myers. Civitas: Toward a secure voting system. In *2008 IEEE Symposium on Security and Privacy (sp 2008)*, pages 354–368. IEEE, 2008.

[21] N. J. Conroy, V. L. Rubin, and Y. Chen. Automatic deception detection: Methods for finding fake news. In *Proceedings of the 78th ASIS&T Annual Meeting: Information Science with Impact: Research in and for the Community*, page 82. American Society for Information Science, 2015.

[22] V. Cortier, D. Galindo, R. Küsters, J. Mueller, and T. Truderung. Sok: Verifiability notions for e-voting protocols. In *2016 IEEE Symposium on Security and Privacy (SP)*, pages 779–798. IEEE, 2016.

[23] V. Costan and S. Devadas. Intel SGX explained. In *Cryptology ePrint Archive, Report 2016/086*, 2016.

[24] V. Costan, I. A. Lebedev, and S. Devadas. Sanctum: Minimal hardware extensions for strong software isolation. In *Proceedings of the 25th USENIX Security Symposium (USENIX Security)*, pages 857–874, 2016.

[25] H. Davies. Ted cruz using firm that harvested data on millions of unwitting facebook users. *The Guardian*, 2015.

[26] M. Del Vicario, A. Bessi, F. Zollo, F. Petroni, A. Scala, G. Caldarelli, H. E. Stanley, and W. Quattrociocchi. The spreading of misinformation online. *Proceedings of the National Academy of Sciences*, 113(3):554–559, 2016.

[27] S. Delaune, S. Kremer, and M. Ryan. Coercion-resistance and receipt-freeness in electronic voting. In *19th IEEE Computer Security Foundations Workshop (CSFW'06)*, pages 12–pp. IEEE, 2006.

[28] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The second-generation onion router. In *Proceedings of the 13th USENIX Security Symposium (USENIX Security)*, 2004.

[29] P. A. Dow, L. A. Adamic, and A. Friggeri. The anatomy of large facebook cascades. 2013.

[30] S. Dziembowski, L. Eckey, and S. Faust. Fairswap: How to fairly exchange digital goods. In *Proceedings of the 25th ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 967–984. ACM, 2018.

[31] M. Egele, G. Stringhini, C. Kruegel, and G. Vigna. Compa: Detecting compromised accounts on social networks. In *Proceedings of the 20th Annual Network and Distributed System Security Symposium (NDSS)*, 2013.

[32] C. Evans, C. Palmer, and R. Sleevi. Public key pinning extension for HTTP. RFC, 2015.

[33] S. Even and Y. Yacobi. Relations among public key signature systems. Technical report, Computer Science Department, Technion, 1980.

[34] Facebook. Working to Stop Misinformation and False News. 2017. https://www.facebook.com/facebookmedia/blog/working-to-stop-misinformation-and-false-news.

[35] R. Ferguson. Word of mouth and viral marketing: taking the temperature of the hottest trends in marketing. *Journal of consumer marketing*, 25(3):179–182, 2008.

[36] M. J. Fischer, N. A. Lynch, and M. S. Paterson. Impossibility of distributed consensus with one faulty process. Technical report, MIT, 1982.

[37] M. Fisher, J. W. Cox, and P. Hermann. Pizzagate: From rumor, to hashtag, to gunfire in dc. *The Washington Post*, 2016.

[38] A. Friggeri, L. A. Adamic, D. Eckles, and J. Cheng. Rumor cascades. In *Proceedings of the International Conference and Workshop on Social Media (ICWSM)*, 2014.

[39] Germany. Strafgesetzbuch, § 107a, Wahlfälschung, 2018. https://dejure.org/gesetze/StGB/107a.html.

[40] Z. Gilani, E. Kochmar, and J. Crowcroft. Classification of twitter accounts into automated agents and human users. In *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, pages 489–496. ACM, 2017.

[41] A. Goyal, F. Bonchi, and L. V. S. Lakshmanan. A data-based approach to social influence maximization. *Proceedings of the VLDB Endowment*, 5, 2011.

[42] J. A. Halderman, S. D. Schoen, N. Heninger, W. Clarkson, W. Paul, J. A. Calandrino, A. J. Feldman, J. Appelbaum, and E. W. Felten. Lest We Remember: Cold-boot

Attacks on Encryption Keys. *Communications of the ACM*, 52(5):91–98, 2009.

[43] M. Hirt and K. Sako. Efficient receipt-free voting based on homomorphic encryption. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 539–556. Springer, 2000.

[44] Intel. Intel SGX, Ref. No.: 332680-002, 2015. https://software.intel.com/sites/default/files/332680-002.pdf.

[45] Intel. SGX SDK, 2016. https://software.intel.com/en-us/sgx-sdk.

[46] Intel. Intel Software Guard Extensions - Developer Zone - Details, 2017. https://software.intel.com/en-us/sgx/details.

[47] N. Jindal and B. Liu. Opinion spam and analysis. In *Proceedings of the 2008 International Conference on Web Search and Data Mining (WSDM)*, pages 219–230. ACM, 2008.

[48] S. Johnson, V. Scarlata, C. Rozas, E. Brickell, and F. Mckeen. Intel SGX: EPID Provisioning and Attestation Services, 2016. https://software.intel.com/en-us/blogs/2016/03/09/intel-sgx-epid-provisioning-and-attestation.

[49] A. Juels, D. Catalano, and M. Jakobsson. Coercion-resistant electronic elections. In *Proceedings of the 2005 ACM workshop on Privacy in the electronic society*, pages 61–70. ACM, 2005.

[50] M. Juuti, B. Sun, T. Mori, and N. Asokan. Stay on-topic: Generating context-specific fake restaurant reviews. In *Proceedings of the European Symposium on Research in Computer Security (ESORICS)*, 2018.

[51] B. Kauer. Oslo: Improving the security of trusted computing. In *Proceedings of the 16th USENIX Security Symposium (USENIX Security)*, pages 229–237, 2007.

[52] D. Kempe, J. Kleinberg, and É. Tardos. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 137–146. ACM, 2003.

[53] P. Koshy, D. Koshy, and P. McDaniel. An analysis of anonymity in bitcoin using p2p network traffic. In *International Conference on Financial Cryptography and Data Security*, pages 469–485. Springer, 2014.

[54] R. Küsters, T. Truderung, and A. Vogt. Accountability: definition and relationship to verifiability. In *Proceedings of the 17th ACM conference on Computer and communications security*, pages 526–535. ACM, 2010.

[55] B. Lee, C. Boyd, E. Dawson, K. Kim, J. Yang, and S. Yoo. Providing receipt-freeness in mixnet-based voting protocols. In *International Conference on Information Security and Cryptology*, pages 245–258. Springer, 2003.

[56] B. Lee and K. Kim. Receipt-free electronic voting scheme with a tamper-resistant randomizer. In *International Conference on Information Security and Cryptology*, pages 389–406. Springer, 2002.

[57] S. Lee, M.-W. Shih, P. Gera, T. Kim, H. Kim, and M. Peinado. Inferring fine-grained control flow inside sgx enclaves with branch shadowing. In *Proceedings of the 26th USENIX Security Symposium (USENIX Security)*, pages 16–18, 2017.

[58] S. Matetic, M. Ahmed, K. Kostiainen, A. Dhar, D. Sommer, A. Gervais, A. Juels, and S. Capkun. ROTE: Rollback protection for trusted execution. In *Proceedings of the 26th USENIX Security Symposium (USENIX Security)*, pages 1289–1306, 2017.

[59] S. Matetic, M. Schneider, A. Miller, A. Juels, and S. Capkun. Delegatee: Brokered delegation using trusted execution environments. In *Proceedings of the 27th USENIX Security Symposium (USENIX Security)*. USENIX Association, 2018.

[60] F. McKeen, I. Alexandrovich, A. Berenzon, C. V. Rozas, H. Shafi, V. Shanbhogue, and U. R. Savagaonkar. Innovative Instructions and Software Model for Isolated Execution. In *HASP ISCA*, 2013.

[61] T. Moran and M. Naor. Receipt-free universally-verifiable voting with everlasting privacy. In *Annual International Cryptology Conference*, pages 373–392. Springer, 2006.

[62] M. Motoyama, K. Levchenko, C. Kanich, D. McCoy, G. M. Voelker, and S. Savage. Re: Captchas-understanding captcha-solving services in an economic context. In *Proceedings of the 19th USENIX Security Symposium (USENIX Security)*, volume 10, page 3, 2010.

[63] A. Mukherjee, B. Liu, and N. Glance. Spotting fake reviewer groups in consumer reviews. In *Proceedings of the 21st International Conference on World Wide Web (WWW)*, pages 191–200. ACM, 2012.

[64] C. B. Mulligan and C. G. Hunter. The empirical frequency of a pivotal vote. *Public Choice*, 116(1):31–54, Jul 2003.

[65] S. Nakamoto. Bitcoin: A peer-to-peer electronic cash system. 2008.

[66] M. Ott, C. Cardie, and J. Hancock. Estimating the prevalence of deception in online review communities. In *Proceedings of the 21st International Conference on World Wide Web (WWW)*, pages 201–210. ACM, 2012.

[67] M. Pennacchiotti and A.-M. Popescu. A machine learning approach to twitter user classification. pages 281–288, 2011.

[68] H. Rainie, J. Q. Anderson, and J. Albright. *The future of free speech, trolls, anonymity and fake news online*. Pew Research Center Washington, DC, 2017.

[69] F. Reid and M. Harrigan. An analysis of anonymity in the bitcoin system. In *Security and Privacy in Social Networks*, pages 197–223. Springer, 2013.

[70] E. B. Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza. Zerocash: Decentralized anonymous payments from bitcoin. In *Proceedings of the 35th IEEE Symposium on Security and Privacy (SP)*, pages 459–474. IEEE, 2014.

[71] S. Senecal and J. Nantel. The influence of online product recommendations on consumers' online choices. *Journal of retailing*, 80(2):159–169, 2004.

[72] C. Shao, G. L. Ciampaglia, O. Varol, K.-C. Yang, A. Flammini, and F. Menczer. The spread of low-credibility content by social bots. *Nature communications*, 9(1):4787, 2018.

[73] K. Shu, A. Sliva, S. Wang, J. Tang, and H. Liu. Fake news detection on social media: A data mining perspective. *ACM SIGKDD Explorations Newsletter*, 19(1):22–36, 2017.

[74] J. Song, S. Lee, and J. Kim. Crowdtarget: Target-based detection of crowdturfing in online social networks. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 793–804. ACM, 2015.

[75] Switzerland. Schweizerisches Strafgesetzbuch, Artikel 279, Störung und Hinderung von Wahlen und Abstimmungen, 2019. https://www.admin.ch/opc/de/classified-compilation/19370083/index.html#a279.

[76] Twitter. Our approach to bots and misinformation. 2017. https://blog.twitter.com/official/en_us/topics/company/2017/Our-Approach-Bots-Misinformation.html.

[77] Twitter. 2018 U.S. midterm elections review. 2019. https://blog.twitter.com/en_us/topics/company/2019/18_midterm_review.html.

[78] United States. United States Code, 2006 Edition, Supplement 5, Title 42 - THE PUBLIC HEALTH AND WELFARE, 2006. https://www.govinfo.gov/app/details/USCODE-2011-title42/USCODE-2011-title42-chap20.

[79] J. Van Bulck, M. Minkin, O. Weisse, D. Genkin, B. Kasikci, F. Piessens, M. Silberstein, T. F. Wenisch, Y. Yarom, and R. Strackx. Foreshadow: Extracting the keys to the intel SGX kingdom with transient out-of-order execution. In *Proceedings of the 27th USENIX Security Symposium (USENIX Security)*, pages 991–1008, 2018.

[80] B. Viswanath, M. A. Bashir, M. Crovella, S. Guha, K. P. Gummadi, B. Krishnamurthy, and A. Mislove. Towards detecting anomalous user behavior in online social networks. In *Proceedings of the 23rd USENIX Security Symposium (USENIX Security)*, pages 223–238, 2014.

[81] J. Vitak, P. Zube, A. Smock, C. T. Carr, N. Ellison, and C. Lampe. It's complicated: Facebook users' political participation in the 2008 election. *CyberPsychology, Behavior, and Social Networking*, 14(3):107–114, 2011.

[82] L. Von Ahn, M. Blum, N. J. Hopper, and J. Langford. Captcha: Using hard ai problems for security. In *International Conference on the Theory and Applications of Cryptographic Techniques (EUROCRYPT)*, pages 294–311. Springer, 2003.

[83] S. Vosoughi, D. Roy, and S. Aral. The spread of true and false news online. *Science*, 359, 2018.

[84] G. Wang, T. Wang, H. Zheng, and B. Y. Zhao. Man vs. machine: Practical adversarial detection of malicious crowdsourcing workers. In *Proceedings of the 23rd USENIX Security Symposium (USENIX Security)*, pages 239–254, 2014.

[85] G. Wang, C. Wilson, X. Zhao, Y. Zhu, M. Mohanlal, H. Zheng, and B. Y. Zhao. Serf and turf: Crowdturfing for fun and profit. In *Proceedings of the 21st International Conference on World Wide Web (WWW)*, pages 679–688. ACM, 2012.

[86] J. Winter. Trusted Computing Building Blocks for Embedded Linux-based ARM Trustzone Platforms. In *Proceedings of the 3rd ACM workshop on Scalable Trusted Computing (STC)*, 2008.

[87] R. Wojtczuk and J. Rutkowska. Attacking SMM Memory via Intel CPU Cache Poisoning. *Invisible Things Lab*, 2009.

[88] K. Wüst, S. Matetic, M. Schneider, I. Miers, K. Kostiainen, and S. Capkun. Zlite: Lightweight clients for shielded zcash transactions using trusted execution. In *International Conference on Financial Cryptography and Data Security*. Springer, 2019.

[89] Y. Yao, B. Viswanath, J. Cryan, H. Zheng, and B. Y. Zhao. Automated crowdturfing attacks and defenses in online review systems. In *Proceedings of the 24th ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 1143–1158. ACM, 2017.

[90] G. Ye, Z. Tang, D. Fang, Z. Zhu, Y. Feng, P. Xu, X. Chen, and Z. Wang. Yet another text captcha solver: A generative adversarial network based approach. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 332–348. ACM, 2018.

[91] Q. Ye, R. Law, B. Gu, and W. Chen. The influence of user-generated content on traveler behavior: An empirical investigation on the effects of e-word-of-mouth to hotel online bookings. *Computers in Human behavior*, 27(2):634–639, 2011.

[92] Zcash Company. bellman, a zk-snark library, 2019. https://github.com/zkcrypto/bellman.

[93] F. Zollo, A. Bessi, M. Del Vicario, A. Scala, G. Caldarelli, L. Shekhtman, S. Havlin, and W. Quattrociocchi. Debunking in a world of tribes. *PloS one*, 12(7):e0181821, 2017.

A Background

A.1 Trusted Execution Environments

Modern TEE environments, such as ARM TrustZone [4, 86] and Intel SGX [23, 44], enable isolated code execution within a user's system with several additional properties. Namely, in this work we use Intel's SGX as the example TEE for implementing TEEVIL. SGX is represented through an instruction set architecture extension for Intel CPUs. The isolated execution is manifested through so called secure *enclaves* that have user-level privileges and which can be attested. Enclaves are accessed using `ocall`/`ecall` interfaces [45] that switch control between the enclaves and the OS. Intel SGX architecture offers protections from a compromised OS, other malicious applications, VMs, BIOS and even other insecure hardware on the residing platform [16, 42, 51, 87]. Below we briefly summarize the main protective mechanisms supported by SGX. Readers familiar with Intel SGX can skip the rest of this subsection. For in-depth background of SGX see [23, 46].

Attestation. Attestation is the process of verifying that enclave code has been properly initialized [23, 48]. A statement containing measurements of enclave's initialization sequence, code, and issuer key is created, signed by the Qouting Enclave and forwarded to the remote verifier which can check the signature using Intel's online attestation service.

Isolation. SGX security architecture guarantees enclave *isolation* [60], using protective mechanisms enforced in the processor, from all software running outside of the enclave. The control-flow integrity of the enclave is preserved and the state is not observable. Additionally, all runtime enclave memory is encrypted and cannot be accessed by the OS.

Sealing. To securely store confidential data across reboots (for persistent storage) by encrypting and authenticating it, enclaves use a mechanism called sealing [5]. Each enclave is provided with a private sealing key (derived from the master Fuse key and Identity Key) that is used for this action.

A.2 Cryptocurrencies

Digital currencies that are based on blockchains became widely used with the rise of *Bitcoin* [65] and are now known under the name cryptocurrencies. To this date there exist many hundreds different blockchain based cryptocurrencies. Such systems allow anyone to issue payments to other parties in a peer-to-peer fashion without any trust assumptions on a single central entity. However, *Bitcoin* is not an ideal replacement for the long-standing cash system because it forfeits user privacy [6, 69]. Instead, it provides pseudonymity, where every user hides behind one or more pseudonyms. Recent work has shown, that pseudonyms in Bitcoin can be linked easily [53]. New systems, such as *ZCash* [70], have been proposed to provide fully anonymous transactions by taking advantage of recent advances in succinct non-interactive zero-knowledge proofs (zk-SNARKs) [8]. These transactions not only hide the participating parties but also the transferred amount while still guaranteeing the correctness of the transaction.

Consistent View of a Blockchain. Given two blockchains A and B, with A being a longer chain than B, we say that A is consistent with B if both A and B are valid blockchains and A is an extension of B, in the sense that it contains at least all the blocks of B in the same order.

A.3 Brokered Delegation

Secure and flexible delegation of credentials and rights for a variety of different service was introduced in [59]. The authors demonstrate the potential of using TEEs for the secure delegation of credentials primarily in the context of payments.

A.4 E-voting

A wide range of security properties are usually looked at when designing an e-voting protocol, notably: (i) several

flavours of verifiability [22,54], (ii) privacy, and (iii) coercion-resistance [49]. First, verifiability (i) comes in different forms, three of whom are common: individual, universal, and end-to-end verifiability. *Individual* verifiability refers to the ability of the voter to check whether its vote has been recorded correctly by the voting authorities, generally this means verifying that the ballot is present in a bulletin board, and that said ballot contains the choice intended by the voter. *Universal* verifiability allows to verify that all the honest votes present in the bulletin board are tallied and counted correctly. *End-to-end* verifiability allows to check that the votes of all the honest voters whom have checked their vote are cast and counted correctly. Second, privacy (ii) should guarantee that the preference of a voter (including whether he or she voted at all) remains confidential. Coercion-resistance (iii) aims to prevent coercion of votes mostly by making it impossible for a voter to prove to a third party that he or she voted in a particular way. Coercion-resistance (iii) is a stronger property than privacy. A common way to achieve (iii) is to require receipt-freeness [43,55,56,61], which ensures that either the voter does not get a confirmation binding him to a specific choice, or that he can fake any such receipt to a third party, thus making it impossible for said third party to rely on the information provided by the voter. While receipt-freeness can be used to obtain coercion-resistance, there is a difference between the two properties [27]. In receipt-freeness the coercer is usually modeled as an observer that can merely examine the transcript of the interaction between the voter and the voting servers, while in coercion-resistance the coercer is interactive and can instruct the voters to reveal private keys and inject chosen messages during the voting process. Note that both receipt-freeness and coercion-resistance cannot be achieved if the e-voting protocol does not guarantee privacy.

B Distributed TEEVIL

It is easy to extend the TEEVIL protocol to allow multiple independent parties to run a service enclave and a payment enclave (cf. Section 3.4). The protocol already supports multiple enclaves running in parallel, and there is no functional or security requirement forcing them to be run only in a single server maintained by a single infrastructure maintainer. To incentivise people to help TEEVIL scale, and be more tolerant to DoS attacks, anyone running one of these enclaves can get a reward for the actions going through their machine. Payment enclaves can send a previously agreed part of the payment to the blockchain address of their owner. Service enclaves need to provide the blockchain address in which they wish to be rewarded to the interface enclave. Their blockchain address can be given to the interface enclave during the enlistment phase (cf. Section 3.4), the interface enclave would then take care of instructing the payment enclaves to issue a reward to the owners of the relevant service enclaves.

Distributing the interface enclave requires a bit more consideration, as we need to define: (i) how to keep a global view of all the users, (ii) how the interactions between different instances of the interface enclaves would work, and (iii) how different instances synchronize with other payment and service enclaves to carry out a campaign. To meet these requirements we envision a system in which multiple interface enclaves manage campaigns independently of each other, but keep a global list of identity owners. With this architecture, each enclave type (interface, payment, and service) is connected to at least another interface enclave, and interface enclaves need to have at least one payment enclave and one service enclave connected to them to be operational. We provide an example of this topology in Figure 5a. This topology allows to easily address (ii) and (iii), since each interface enclave can operate independently of the others, therefore no synchronization or communication is required for a campaign creation (cf. Section 3.2.2) and during the automatic interactions (cf. Section 3.2.3) phase. Decentralizing TEEVIL in this way implies that no change is required for the protocol of these two phases, besides the detail that instead of contacting the infrastructure maintainer, now the identity renter contacts any interface enclave, and that interface enclave takes care of completing the campaign. To ensure that funds are not lost if an interface enclave is killed in the middle of a campaign, any payment enclave which was part of the campaign could inform another interface enclave and coordinate with it to stop the campaign and return the remaining funds.

Regarding (i), we observe that interface enclaves do not need to keep a perfectly synchronized global view of all the users enrolled. Therefore in this design, it would be sufficient to employ a gossip protocol in which, in each round, every interface enclave lets its neighbouring interface enclaves know about new credentials. As long as the network of interface enclaves is not partitioned this will ensure that eventually each interface enclave will be aware of a new user. To not flood the network this could be done in batches of users. If a group of interface enclaves goes offline before their identity owners were propagated to any other node, we can simply let the lost identity owners re-enroll with a new interface enclave. However, this has the downside that if any identity owner wants to update their policies, they will not be instantly reflected in the whole network.