# BISCUIT - Blockchain Security Incident Reporting based on Human Observations

Benedikt Putz
benedikt.putz@ur.de
University of Regensburg
Regensburg, Germany

Manfred Vielberth
manfred.vielberth@ur.de
University of Regensburg
Regensburg, Germany

Günther Pernul
guenther.pernul@ur.de
University of Regensburg
Regensburg, Germany

## ABSTRACT

Security incidents in blockchain-based systems are frequent nowadays, which calls for more structured efforts in incident reporting and response. To improve the current status quo of reporting incidents on blogs and social media, we propose a decentralized incident reporting and discussion system. Our approach guides users (security novices) towards a classification of their observations using a tiered taxonomy of blockchain incidents. Questions based on previous incidents interactively support the classification. Post submission a security incident response committee then discusses these observations on our decentralized platform to decide on an appropriate response. For evaluation, we implement our model as a decentralized application and demonstrate its practical suitability in a preliminary user study.

## CCS CONCEPTS

• **Security and privacy → Intrusion detection systems**; **Distributed systems security**; **Domain-specific security and privacy architectures**.

## KEYWORDS

Human-as-a-Security-Sensor, Blockchain, Distributed Ledger, Incident Detection

**ACM Reference Format:**
Benedikt Putz, Manfred Vielberth, and Günther Pernul. 2022. BISCUIT - Blockchain Security Incident Reporting based on Human Observations. In *The 17th International Conference on Availability, Reliability and Security (ARES 2022), August 23–26, 2022, Vienna, Austria.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3538969.3538984

## 1 INTRODUCTION

Despite the widely accepted notion that blockchain systems are secure, security incidents are widespread in permissionless blockchains today. The impact is often measured in millions of dollars, especially when Decentralized Finance (DeFi) applications are targeted. Early famous examples include the DAO attack in 2016 and the Parity Multisig wallet hack in 2017 [3]. With the exponential growth of cryptocurrency market capitalization and the emergence of DeFi, attack frequency has increased enough to warrant weekly newsletters on the latest incidents [10]. While permissionless cryptocurrency-based blockchains are the primary subject of media coverage, similar security issues apply to permissioned blockchains [5, 19]. Permissioned blockchains often secure high-value real-world assets like trade-finance transactions or freight, making them an attractive target for attackers.

Still, current reporting of blockchain security incidents is unstructured and centralized in the form of tweets or blog posts [10]. Structured threat intelligence is hard to find, and security professionals have to scrape together information from various sources. To facilitate targeted responses, structured reporting and incident response are needed, adapted to decentralized stakeholders. Current blockchain engineering efforts are focused on application development, not security incidents. Research can help by structuring threats found in the literature and making them more accessible.

Blockchain security incidents are challenging to handle due to the distributed nature of the blockchain, which makes threats hard to attribute, reconstruct and reproduce [3]. The layered structure of blockchains also offers many points of attack on the protocol, networking and application layers [20]. In practice, blockchain attacks are often initially discovered by users alerting developers towards problems or inconsistencies [10]. However, finding the contact information can be difficult as few DApps have consistent and transparent security reporting policies. Even if it is found, users have no guidance on how to report their observations in a structured fashion. Commonly used responsible disclosure standards focus on disclosure of vulnerabilities, not security issues in general [15].

Despite some efforts to detect vulnerabilities before they are exploited [3], the majority of blockchain incident reporting is still done on social media and blog sites [10]. This work proposes a novel approach based on the Human-as-a-Security-Sensor (HaaSS) paradigm [7]. We state the following research questions:

*RQ1.* Can human observations support the detection of blockchain security incidents?

*RQ2.* How can the incident response process for human-reported incidents be structured and made tamper-proof?

Based on these research questions, we follow the design science research methodology [18] to develop a decentralized blockchain security incident reporting and discussion system. The system focuses on recording human observations of (suspected) blockchain security incidents and their subsequent enrichment and discussion by concerned parties. Metadata of the discussion is recorded on-chain, ensuring transparency in the incident response process. It also prevents tampering by attackers, which would be possible when using out-of-band communication channels.

*ARES 2022, August 23–26, 2022, Vienna, Austria*
© 2022 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-9670-7/22/08.
https://doi.org/10.1145/3538969.3538984

*Contributions.* This work focuses on providing a structured process for users to report observations of blockchain threats. In summary, we provide the following contributions to research:

- a process for integrating human observed incidents in smart contract incident response
- a taxonomy for blockchain security threats for use in incident reporting
- a decentralized incident discussion model for incident response without a central decision-making entity

The paper is structured as follows. We first provide some background on reporting human observations and blockchain threats in Section 2. Thereafter, we propose our model BISCUIT in Section 3 by introducing preliminaries and a formal description. Section 4 elaborates on our taxonomy for reporting observations, consisting of threats and threat indicators. Section 5 then introduces the technical architecture to support the incident reports, along with a description of our implementation on Ethereum. We proceed to evaluate the implementation with a user study in Section 6 and discuss the results in Section 7.

## 2 BACKGROUND

*The HaaSS paradigm.* In information security, there have been approaches for a while that try to involve humans in incident detection. For example, almost all email systems provide the possibility to report spam and phishing emails. However, this has only been done for specific problems, and humans have continued to be seen primarily as a problem. However, this view has become outdated, and there is a paradigm shift from human-as-a-problem to human-as-a-solution [31]. The human-as-a-security-sensor paradigm emerged, attempting to actively involve humans into security [7, 28]. Especially for security incidents with limited technical traces, one has to rely on information provided by humans. However, people find it challenging to record security incidents in a structured way, especially if they are not security experts. Therefore, it is necessary to provide assistance that provides structure [27].

*Blockchain threat detection.* In blockchain research and practice, most attention has been given exploitation of smart contract vulnerabilities [3]. Tool-based automated approaches for detecting and mitigating common vulnerability classes include proactive defenses to prevent attacks, and reactive defenses for previously unknown or hidden vulnerabilities. Proactive defenses include specific languages focused on security, programming best practices, vulnerability scanners and hardening measures applied to the blockchain itself or the smart contract. Reactive defenses aim to prevent exploitation by verifying execution results at contract runtime.

In practice, the main precautions being taken are vulnerability scans, security audits and bug bounty programs. However, these measures must be implemented before deploying a smart contract and may miss novel vulnerabilities. This is evident from numerous attacks in the Decentralized Finance (DeFi) ecosystem, which are not only related to software bugs, but require deep understanding of DeFi business logic. Detecting such attacks is difficult with automated systems [3, 29]. Early tool-based approaches for detection like BlockEye [29] use rule-based systems to flag suspicious transactions. However, these systems may miss attacks that are not caught by rules or generate a significant amount of false positives.

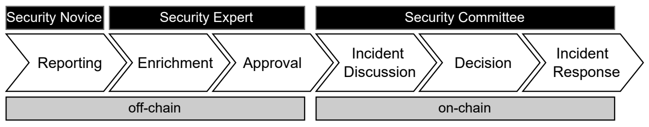

**Figure 1: Roles and process steps for incident reporting**

## 3 THE BISCUIT MODEL

This Section introduces our conceptual approach and formal model for the BISCUIT prototype (**B**lockcha**i**n **Secu**rity **I**ncident Repor**t**ing). We begin by stating our goals and establishing some definitions in Section 3.1.

### 3.1 Preliminaries

*3.1.1 Goal.* Ideally, the goal of detecting an attack would be to stop it and prevent any malicious consequences. However, few attacks on smart contracts can be prevented through reactive defense [3]. The near-immediate and immutable effects of a blockchain transaction make it difficult to reverse attacks. Hence, the main goals of incident detection are damage minimization (protecting user funds), accountability and forensic intelligence.

*3.1.2 Requirements.* To ensure proper support of incident detection by human observations (RQ1), we state several requirements. To ensure that users are able to report their observations, the model must be *learnable.* In addition, in a decentralized network with semi-trusted participants reporting must be *transparent* and tamper-proof (*integrity*-preserving). In summary we require:

**Learnability.** Users should be able to accomplish their tasks the first time they use the incident reporting tool.
**Transparency.** Anyone with access to the affected blockchain should be able to see and verify incident reports.
**Integrity.** Incident reports should be protected from unauthorized tampering (modification, false data injection).

*3.1.3 Roles.* There are three types of participants involved in the incident reporting and response process. Security Novices and Security Experts represent a well-known distinction from literature [1] in terms of corporate security. The Security Committee is introduced as a collective term for all Security Experts responsible for security incident handling.

**Security Novices** are users who do not necessarily have deep security knowledge. Any DLT user can become a Security Novice by interacting with BISCUIT. We assume some basic knowledge regarding blockchain concepts, meaning the user is able to understand block and transaction details shown on a blockchain explorer.

**Security Experts** are users who have more in-depth security knowledge and therefore a deeper understanding and a broader view of security issues in DLT. An expert can evaluate if a report is an incident, and whether it should be shared with other experts in the Security Committee. He therefore has the necessary permissions to publish security incidents reported by novices.

The **Security Committee** is a set of Security Experts responsible for handling reported security incidents. The aim of the Security Committee is to make a decision regarding an appropriate response. Who constitutes the committee is application and context-dependent. Sensible choices could be administrators of a permissioned blockchain's nodes, or for permissionless blockchains the shareholders of a decentralized application or Decentralized Autonomous Organization (DAO).

## 3.2 Formal model

In order to support incident reporting, it is first necessary to capture which elements of an incident can be observed by humans and therefore reported. A frequently used model for this is provided by NIST [9]. In a similar form, the model has already been used in the context of the human-as-a-security-sensor paradigm [27] and is also used in incident reporting, though in a more complex form and thus not suitable for the use case in this work [14]. The goal of the approach is to capture the information of the incident as structured as possible. As presented in prior work [28], this enables an automated conversion into a common incident reporting format such as STIX[1]. Derived from this, a normalized incident consists of four binary vectors, each representing the elements that can be reported by a Security Novice: Sources $\vec{s}$ that trigger the incident, events that occur during the incident $\vec{e}$, affected entities $\vec{a}$ that are influenced negatively by the incident, and an impact $\vec{i}$ that indicates how serious the incident is considered to be by the Security Novice. Thus, the incident results from the concatenation of the vectors:

$$\overrightarrow{inc} = \vec{s} \frown \vec{e} \frown \vec{a} \frown \vec{i}$$

## 3.3 Incident Reporting Flow

The Incident Reporting Flow (see Figure 1) aims to provide a way to report security incidents in a DLT in a structured way, and to find an appropriate response to them without relying on a central decision point. It consists of six steps executed by different roles, with the first three steps executed off-chain. A detailed description of the process steps is given in the following paragraphs.

*Reporting.* The process begins when the user of the DLT notices an event that he considers to be a security incident. The process itself is then triggered by the user reporting this security incident. To enable the reporting process to be as structured and complete as possible, the user is supported in two stages.

In the first stage, possible security incidents or classes of security incidents that can occur on a DLT are provided in the form of a taxonomy. With the help of the taxonomy, the user can work his way down from very general classes to very specific security incidents without losing the overview. Each element of the taxonomy corresponds to one element of the binary incident vector $\overrightarrow{inc}$. Thus the result of this stage is an instance of the incident vector ($\overrightarrow{INC}_{init}$) with each element selected by the user with a value of 1.

The next stage of reporting is based on the assumption that security incidents in many cases have some similarity to past security incidents. Thus, similar to a recommender system [12, 30] (instead of similar products, similar incidents are suggested) the user can be assisted in completing or correcting the reported security incident. The reported data is compared with past incidents ($\overrightarrow{INC}_{past}$) by calculating the similarity of the reported incident and each past incident in a database containing n incidents:

$$sim(\overrightarrow{INC}_{init}, \overrightarrow{INC}_{past,i}) \qquad i \in [0, n]$$

[1]oasis-open.github.io/cti-documentation/

The questions are generated based on the differences to the most similar incidents. By answering these questions, the user can refine or correct the data, which results in a corrected version of the reported incident $\overrightarrow{INC}_{corr}$.

*Enrichment & Approval.* In this phase, the local organization's Security Expert receives the report and contributes additional information from the perspective of a privileged user. Due to elevated or administrative privileges the potential security insight can be enriched with deeper insight or discarded if it is only a temporary technical issue. Otherwise, the Security Expert approves the incident to publish it to the entire Security Committee. Thereby, he validates that the report does not contain any sensitive information and thus does not violate company policies.

*Incident Discussion.* Published incidents are available to the Security Committee. Each member can contribute their view on the incident through authenticated comments $c$. Besides viewing the structured incident data provided by the incident issuer, Security Experts can add their own intelligence through attachments. They can also propose an update to the now shared incident. This update remains a proposal until a configurable threshold $t_{vote}$ of upvotes for the proposal has been passed, at which point the original incident reference is replaced as part of the voting transaction. Similarly, users can propose to update the default status of the incident to one of the following states: in discussion ($S_0$), response initiated ($S_1$), invalid ($S_2$), duplicate ($S_3$), and completed ($S_4$). Status updates must be approved with the same voting threshold $t_{vote}$.

*Decision & Incident Response.* Finally, the Security Committee must come to a consensus on how to deal with the incident. Based on the jointly provided evidence, the Committee may decide to discard the incident, or to initiate an incident response action. To suggest a decision, any Expert may submit a response comment $cr_i$ to the incident. This comment may contain multiple response actions $ar_{ij}$. The committee can vote on $cr_i$ to approve $ar_{ij} \forall j \in J$. If a configurable threshold $t_{vote}$ is passed, the response actions of $ar_i$ are considered approved.

## 4 INCIDENT TAXONOMY

To enable structured reporting of security incidents within the smart contracts, a structured collection of possible incidents is necessary. A taxonomy is particularly suitable for addressing the problem, that security novices might not have a very detailed knowledge about observed incidents as it allows the novice to work his way from a very abstract to an increasingly fine-grained view. For

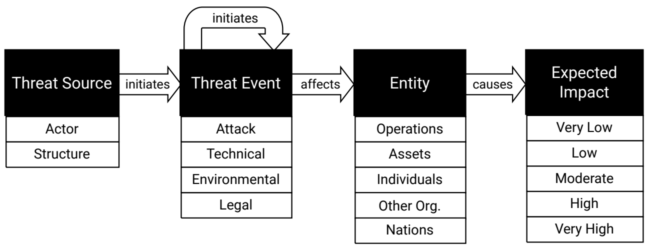

**Figure 2: Abstract structure of the taxonomy [28].**

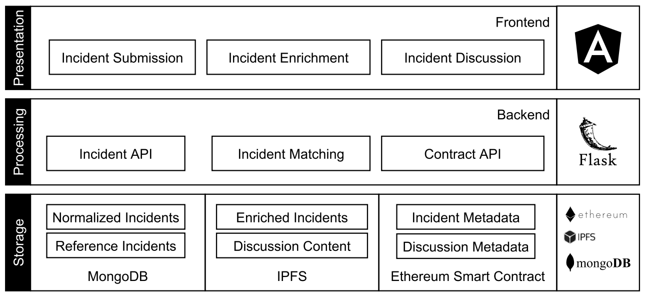

**Figure 3: Prototype layers and modules.**

this purpose, an existing taxonomy for security incidents [28] is extended with blockchain-specific incidents. The first layer of the taxonomy can be seen in Figure 2, while the complete taxonomy is published in the GitHub repository[2].

For the extension of the taxonomy, the extended taxonomy design process of Kundisch et al. [11] is applied. This process enhances the prevalent methodology of Nickerson et al. [17], integrating it with the design science research method of Peffers et al. [18].

*Threat Sources.* At the lowest level of the taxonomy, a distinction can be made between actors and structures, which can also be the cause of an incident in the blockchain environment. According to Putz and Pernul [21], five different roles or classes of actors can be distinguished in a blockchain system: Certificate Authority (CA) Admin, Orderer Admin, Peer Admin, Transactor, and External User. To enable in-depth reporting of incident sources in a blockchain environment, we assign them to appropriate nodes in the original taxonomy. In addition to individual actors, software can be the source of a security incident. According to Shirvas et al. [25], two main types can be distinguished in the blockchain environment: decentralized applications (dApps) and smart contracts. There is currently no suitable class in the existing taxonomy, which is why the category Blockchain-Specific Application has been added.

*Threat Events.* To build our taxonomy of threat events $\vec{e}$, we aggregate vulnerability information from various sources obtained in the literature review. Rameder conducts an extensive classification of smart contract vulnerabilities [22], based on a number of other works such as the DASP TOP 10 [16], the Smart Contract Weakness Classification Registry [26] and other surveys [3]. We also include other types of attacks not specific to smart contracts based on relevant literature [5, 13, 21, 23, 24].

## 5 PROTOTYPE

To evaluate the model, we construct a prototype artifact for participants to interact with. First, we propose a generalized architecture for any blockchain based on the current state of the art for decentralized applications in Section 5.1. We further describe our instantiation for Ethereum smart contract incidents in Section 5.2.

### 5.1 Architecture

The architecture of our framework is shown in Figure 3 and described hereafter. As detailed in Section 3, incident handling is initiated by a *Security Novice* or *Expert*. They may submit an incident report using a guided interactive form. This off-chain phase is supported by a website frontend. The client implements the *Incident Reporting* process that guides Security Novices towards submitting a structure incident report. The reported data is sent to a server application supported by a local database for intermediate storage. The incident is stored in original and normalized form. After submission the user is then prompted with several questions, which are used to determine similar incidents.

The local database serves as intermediate storage for Novice-submitted incidents. If the local Expert deems the incident report valid, they may add additional information using the frontend based on own knowledge or data only available to privileged users, such as blockchain node logs. Otherwise, incidents can be discarded if they are not relevant. This might be the case if the user has mistaken a scheduled downtime as a security incident.

Valid incidents are approved by the Expert for further discussion with the Security Committee. They are also added to the local storage to serve as reference incidents for future reports. After approval, the incident reference is published to a smart contract, thus proceeding to the on-chain phase. The actual incident data associated with the reference is transferred to the shared storage.

The on-chain phase consists of a comment-based discussion among Security Experts. Each expert can add comments, optionally with attachments, status updates or an updated incident description. Up- and downvoting incidents serves as an indicator for relevance and allows reaching consensus on status and incident updates.

### 5.2 Implementation

We implement the prototype using a three layer approach: a *Presentation*, *Processing* and *Storage* layer. Figure 3 details the implementation of each layer.

The *Presentation* layer supports both Security Novices and Security Experts in submitting, enriching and discussing incidents. It is implemented as an Angular single page application.

The *Processing* layer provides APIs for the frontend to interact with the *Storage* layer. Besides incident reporting, this includes incident matching, which is based on cosine similarity. The backend is built using Python for its libraries nltk[3] (to generate interactive questions) and numpy[4] (for cosine similarity). The REST API endpoints are built using the Flask[5] library.

The *Storage* layer persists incident and discussion data. The local storage is implemented using MongoDB and stores incident data as JSON documents in original and normalized form. Once original incidents are enriched by the Security Expert and approved, they are moved to a separate collection as *Reference Incidents*. Future user submissions are compared to these reference incidents to improve incident reporting quality. On approval, incident data is also published to the shared storage. The shared storage is implemented using the distributed hash table IPFS[6], a commonly used off-chain DHT database solution. The IPFS reference is used to publish the incident metadata to the Incident Registry smart contract. Our smart contract is based on Solidity, which is currently the the most widely

[2]github.com/biscuitsecurity

[3]nltk.org
[4]numpy.org
[5]flask.palletsprojects.com
[6]ipfs.io

studied language in terms of security vulnerabilities due to the popularity of Ethereum in research and practice [3]. The contract can be deployed on any blockchain supporting Solidity. It is recommended for the incident reporting blockchain to be different from the application blockchain to avoid vulnerabilities common to both blockchains.

The full source code of our implementation is available in multiple GitHub repositories[2].

## 6 EVALUATION

To determine if our model and prototype meet the goals of research questions RQ1 and RQ2, we perform a preliminary user study with Security Novices and Security Experts. The main goal of the user study is to determine the suitability of our prototype form and underlying taxonomy for reporting common threats.

*Setting.* We include incidents in two different settings. The first one is a permissioned blockchain for logistics tracking, similar to TradeLens. Users are tasked to update logistics information, but face four separate issues when attempting to do so. The second scenario is set in the Ethereum DeFi ecosystem. Participants are asked to review transactions by a single account to determine applicable threat indicators. In total each subject performs six incident reports.

*Participants.* Before selecting participants, we consult the university's ethical commission regarding ethical considerations[7]. The consultation yields no concerns as our study does not interview vulnerable groups or perform stressful activities. Additionally, nothing is recorded and no sensitive or personal data are processed. To evaluate **incident reporting**, we interview seven participants that have studied and used blockchain applications, but are not experts in blockchain security. We asked participants to rate their own knowledge regarding blockchain on a Likert scale from 1-7, yielding $\mu = 3.7, \sigma = 1.1$. For blockchain security knowledge, the self-assessment results were slightly worse at $\mu = 2.6, \sigma = 1.3$. Only four of the interviewees had previously used EtherScan. These results confirm that the interviewees are indeed Security Novices.

*Procedure.* The permissioned blockchain attack scenario includes a website outage, identity compromise, consensus algorithm failure and a sybil attack creating fake identities. For the permissionless scenario, users are provided with the addresses of two real attacks from the public Ethereum blockchain. Both attacks are flash loans abusing vulnerabilities in DeFi applications. Users can inspect the transactions via the EtherScan Blockchain Explorer[8]. The full interview guideline with attack descriptions is available on GitHub[2].

*Results.* Interviews took on average 60 minutes including task explanations. We observe a slight negative correlation ($\rho = -0.55, p = 0.15$) regarding interview duration and self-assessed blockchain knowledge, although not statistically significant at our sample size.

As threat sources, the novices identified outsiders and insiders reliably in the taxonomy. For events, based on the available information the users distinguished appropriately between technical issues and nefarious activity. For flash loan attacks, users reliably reported transaction frequency spikes, failed transactions and high gas usage. Regarding affected entities, participants successfully identified the correct taxonomy elements, although some preferred to note assets and others individuals as an affected entity. Finally, impact assessments were at most 2 points apart on our 5 point scale, indicating good judgement.

We also received useful suggestions towards improving the usability of the prototype. For example, users noted that it was sometimes difficult to find the desired entry in the taxonomy. We improved this navigability issue by adding tooltips for individual leaves in the taxonomy tree, highlighting their child nodes.

[7]https://go.ur.de/xGUq3ESN
[8]etherscan.io

## 7 DISCUSSION

We revisit our research questions and discuss whether our prototype and evaluation results support their fulfillment.

*RQ1.* Can human observations support the detection of blockchain security incidents?

The preliminary prototype and user study demonstrate that users of varying knowledge levels are able to contribute to reporting of real blockchain security incidents. Practical sample incidents were reported in a manner that allowed security experts to investigate the incident in more detail. The taxonomy and approval process may affect the timeliness of the incident report, but helps structure threats and prevent spam.

*RQ2.* How can the incident response process for human-reported incidents be structured and made tamper-proof?

Structured reporting of incidents is enabled by the hierarchical taxonomy used during incident reporting. The resulting JSON document can later be transformed into applicable CTI formats such as STIX [28]. Tampering prevention is enabled by storing incident metadata on the blockchain.

## 8 RELATED WORK

The first category of related work focuses on classification of blockchain security threats, while the second category concerns proactive and reactive defenses for integrating humans as part of the blockchain incident security response process.

Many researchers have worked on taxonomies specific to blockchain threats [6, 13] and smart contract threats [3]. A smaller number of works classifies threats specific to permissioned blockchains [5, 19, 21]. The important role of humans in threat mitigation has been recognized with regard to humans acting as developers and users of DLT [6]. Social engineering attacks target human users in particular. As a result, humans are in the best position to report such incidents. Several such attacks have been found to be relevant to Ethereum smart contracts [8]. The authors state that fully automated detection of such attacks is impossible, as human judgement is needed to understand smart contract semantics. This further supports the need for structured reporting of human observations, augmenting and complementing automated tool reports.

Proactive defenses of blockchain threats include humans as part of software audits and bug bounty programs. Researchers have studied blockchain-based bug bounty programs to incentivize white hats to report information about vulnerabilities in smart contracts [2]. While conceptually similar, our work focuses on post-exploitation incident response, whereas bug bounty programs target pre-exploitation vulnerability reports.

Reactive defenses involve alerting humans, which act as security analysts to mitigate ongoing threats. Current research includes online transaction monitoring [4] and visualizations for monitoring, tailored to blockchain security [19]. However, a discussion-based reactive approach to decide on a response for arbitrary incidents has not been proposed so far.

## 9 CONCLUSION

This research paper proposes a novel decentralized approach for reporting blockchain security incidents. The BISCUIT model focuses on learnability for Security Novices, and integrity and transparency for Security Experts. We implement a prototype using a flexible three-layer approach, combining local and shared components to support multiple application contexts. The evaluation results show that it is able to assist users with reporting blockchain incidents. The reports provide structured data, which serves as the basis for incident mitigation discussion. We consider specific mitigation recommendations a subject for future work. Another vital point for future work is a more detailed discussion of system governance, which is essential for scalable and sustainable incident handling. Similarly, a detailed incentivization model for participating novices could enhance the sustainability of the system as part of future work. We conclude that our prototype provides a good starting point for structuring user observations, showing a path forward towards structured threat intelligence for blockchain incidents. Like any other research paper, our work has its weaknesses. It should be mentioned that the user study needs to be conducted with more participants. Above all, the perspectives of all stakeholders should be taken into account to enable an all-encompassing evaluation.

## REFERENCES

[1] Fabian Böhm, Manfred Vielberth, and Günther Pernul. 2021. Bridging Knowledge Gaps in Security Analytics. In *Proceedings of the 7th International Conference on Information Systems Security and Privacy.* SCITEPRESS - Science and Technology Publications, 98–108. https://doi.org/10.5220/0010225400980108

[2] Lorenz Breidenbach, Philip Daian, Florian Tramèr, and Ari Juels. 2018. Enter the Hydra: Towards Principled Bug Bounties and Exploit-Resistant Smart Contracts. In *27th USENIX Security Symposium, USENIX Security 2018, Baltimore, MD, USA, August 15-17, 2018.* USENIX Association, 1335–1352. https://www.usenix.org/conference/usenixsecurity18/presentation/breindenbach

[3] Huashan Chen, Marcus Pendleton, Laurent Njilla, and Shouhuai Xu. 2020. A Survey on Ethereum Systems Security: Vulnerabilities, Attacks, and Defenses. *ACM Comput. Surv.* 53, 3 (2020), 67:1–67:43. https://doi.org/10.1145/3391195

[4] Ting Chen, Rong Cao, Ting Li, Xiapu Luo, Guofei Gu, Yufei Zhang, Zhou Liao, Hang Zhu, Gang Chen, Zheyuan He, Yuxing Tang, Xiaodong Lin, and Xiaosong Zhang. 2020. SODA: A Generic Online Detection Framework for Smart Contracts. In *27th Annual Network and Distributed System Security Symposium, NDSS 2020, San Diego, California, USA, February 23-26, 2020.* The Internet Society, 1–17.

[5] Ahaan Dabholkar and Vishal Saraswat. 2019. Ripping the Fabric: Attacks and Mitigations on Hyperledger Fabric. In *Applications and Techniques in Information Security - 10th International Conference, ATIS 2019, Thanjavur, India, November 22-24, 2019, Proceedings (Communications in Computer and Information Science, Vol. 1116).* Springer, 300–311. https://doi.org/10.1007/978-981-15-0871-4_24

[6] Tobias Guggenberger, Vincent Schlatt, Jonathan Schmid, and Nils Urbach. 2021. A Structured Overview of Attacks on Blockchain Systems. In *25th Pacific Asia Conference on Information Systems, PACIS 2021, Virtual Event / Dubai, UAE, July 12-14, 2021.* 100. https://aisel.aisnet.org/pacis2021/100

[7] Ryan Heartfield and George Loukas. 2018. Detecting semantic social engineering attacks with the weakest link: Implementation and empirical evaluation of a human-as-a-security-sensor framework. *Computers & Security* 76 (2018), 101–127. https://doi.org/10.1016/j.cose.2018.02.020

[8] Nikolay Ivanov, Jianzhi Lou, Ting Chen, Jin Li, and Qiben Yan. 2021. Targeting the Weakest Link: Social Engineering Attacks in Ethereum Smart Contracts. In *ASIA CCS '21: ACM Asia Conference on Computer and Communications Security, Virtual Event, Hong Kong, June 7-11, 2021.* ACM, 787–801. https://doi.org/10.1145/3433210.3453085

[9] Joint Task Force Transformation Initiative. 2012. *Guide for conducting risk assessments.* National Institute of Standards and Technology, Gaithersburg, MD. https://doi.org/10.6028/NIST.SP.800-30r1

[10] Peter Kacherginsky. 2022. Blockchain Threat Intelligence. https://www.blockthreat.io/.

[11] Dennis Kundisch, Jan Muntermann, Anna Maria Oberländer, Daniel Rau, Maximilian Röglinger, Thorsten Schoormann, and Daniel Szopinski. 2021. An Update for Taxonomy Designers. *Business & Information Systems Engineering* (Oct. 2021), 1–19. https://doi.org/10.1007/s12599-021-00723-x

[12] Jie Lu, Dianshuang Wu, Mingsong Mao, Wei Wang, and Guangquan Zhang. 2015. Recommender system application developments: A survey. *Decision Support Systems* 74 (2015), 12–32. https://doi.org/10.1016/j.dss.2015.03.008

[13] Yassine Maleh, Mohammad Shojafar, Mamoun Alazab, and Imed Romdhani. 2020. *Blockchain for cybersecurity and privacy: architectures, challenges, and applications.* CRC Press.

[14] Florian Menges and Günther Pernul. 2018. A comparative analysis of incident reporting formats. *Computers & Security* 73 (2018), 87–101. https://doi.org/10.1016/j.cose.2017.10.009

[15] Darren Mills. 2021. A Standard for Responsible Disclosure in Cryptocurrency and Related Software. RD-Crypto-Spec.

[16] NCC Group. 2018. Decentralized Application Security Project (or DASP) Top 10 of 2018. https://dasp.co/.

[17] Robert C. Nickerson, Upkar Varshney, and Jan Muntermann. 2013. A method for taxonomy development and its application in information systems. *European Journal of Information Systems* 22, 3 (2013), 336–359. https://doi.org/10.1057/ejis.2012.26

[18] Ken Peffers, Tuure Tuunanen, Marcus A. Rothenberger, and Samir Chatterjee. 2007. A Design Science Research Methodology for Information Systems Research. *Journal of management information systems* 24, 3 (2007), 45–77. https://doi.org/10.2753/MIS0742-1222240302

[19] Benedikt Putz, Fabian Böhm, and Günther Pernul. 2021. HyperSec: Visual Analytics for Blockchain Security Monitoring. In *ICT Systems Security and Privacy Protection - 36th IFIP TC 11 International Conference, SEC 2021, Oslo, Norway, June 22-24, 2021, Proceedings (IFIP Advances in Information and Communication Technology, Vol. 625).* Springer, 165–180. https://doi.org/10.1007/978-3-030-78120-0_11

[20] Benedikt Putz and Günther Pernul. 2019. Trust Factors and Insider Threats in Permissioned Distributed Ledgers - An Analytical Study and Evaluation of Popular DLT Frameworks. *Trans. Large Scale Data Knowl. Centered Syst.* 42 (2019), 25–50. https://doi.org/10.1007/978-3-662-60531-8_2

[21] Benedikt Putz and Günther Pernul. 2020. Detecting Blockchain Security Threats. In *2020 IEEE International Conference on Blockchain (Blockchain).* IEEE, 313–320. https://doi.org/10.1109/Blockchain50366.2020.00046

[22] Heidelinde Rameder. 2021. *Systematic Review of Ethereum Smart Contract Security Vulnerabilities, Analysis Methods and Tools.* Diploma Thesis. TU Wien. https://doi.org/10.34726/hss.2021.86784

[23] Muhammad Saad, Jeffrey Spaulding, Laurent Njilla, Charles Kamhoua, Sachin Shetty, DaeHun Nyang, and Aziz Mohaisen. 2019. Exploring the Attack Surface of Blockchain: A Systematic Overview. https://doi.org/10.48550/arXiv.1904.03487

[24] Gjorgji Shemov, Borja Garcia de Soto, and Hoda Alkhzaimi. 2020. Blockchain applied to the construction supply chain: A case study with threat model. *Frontiers of Engineering Management* 7, 4 (2020), 564–577. https://doi.org/10.1007/s42524-020-0129-x

[25] Mahendra Kumar Shrivas, Thomas Yeboah Dean, and S. Selva Brunda. 03.01.2020 - 05.01.2020. The Disruptive Blockchain Security Threats and Threat Categorization. In *2020 First International Conference on Power, Control and Computing Technologies (ICPC2T).* IEEE, 327–338. https://doi.org/10.1109/ICPC2T48082.2020.9071475

[26] SWC. 2022. Smart Contract Weakness Classification and Test Cases. http://swcregistry.io/.

[27] Manfred Vielberth, Ludwig Englbrecht, and Günther Pernul. 2021. Improving data quality for human-as-a-security-sensor. A process driven quality improvement approach for user-provided incident information. *Information & Computer Security* 29, 2 (2021), 332–349. https://doi.org/10.1108/ICS-06-2020-0100

[28] Manfred Vielberth, Florian Menges, and Günther Pernul. 2019. Human-as-a-security-sensor for harvesting threat intelligence. *Cybersecurity* 2, 23 (2019), 1–15. https://doi.org/10.1186/s42400-019-0040-0

[29] Bin Wang, Han Liu, Chao Liu, Zhiqiang Yang, Qian Ren, Huixuan Zheng, and Hong Lei. 2021. BLOCKEYE: Hunting for DeFi Attacks on Blockchain. In *43rd IEEE/ACM International Conference on Software Engineering: Companion Proceedings, ICSE Companion 2021, Madrid, Spain, May 25-28, 2021.* IEEE, 17–20. https://doi.org/10.1109/ICSE-Companion52605.2021.00025

[30] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2019. Deep Learning Based Recommender System: A Survey and New Perspectives. *Comput. Surveys* 52, 1, Article 5 (Feb. 2019), 38 pages. https://doi.org/10.1145/3285029

[31] Verena Zimmermann and Karen Renaud. 2019. Moving from a "human-as-problem" to a "human-as-solution" cybersecurity mindset. *International Journal of Human-Computer Studies* 131 (2019), 169–187. https://doi.org/10.1016/j.ijhcs.2019.05.005