Threat Modeling on an AI Blockchain System: OpenMined

Nico Amstätter-Zöchbauer, Aleksandr Aronikov, Gerhard Klager, and Iris Kompatscher

Abstract—**Building and maintaining secure and privacy-preserving systems is among the key challenges to be considered in modern software development projects. In highly sensitive industries, like healthcare or finance, cyberattacks can have particularly devastating effects on businesses and customers alike, as has been shown in the recent attacks on cryptocurrency exchanges. In this paper, we examine the security and privacy properties of OpenMined, an open-source blockchain system aimed at enabling users to train AI models without the need to obtain a copy of the data used. Our analysis focuses on modeling the potential threats and vulnerabilities by combining various threat modeling methods. As a result, we present several models depicting the system's properties and extracting potential threats. Based on our analysis, we estimate the possible effects of potential attacks, the accompanying privacy implications, and give an outlook on how these shortcomings could be resolved.**

Index Terms—**AI, Blockchain, OpenMined, Threat Modeling**

I. Introduction

Creating systems and applications that are secure and robust against potential cyberattacks is among the main concerns in the development process. The core questions, thus, are "How one can create a system with minimum vulnerabilities to attacks?" as well as "How to cope with the threats of the environment in which the application already exists?". An attempt to solve these issues is threat modeling, the key principles of which rely on finding security issues early in the requirements engineering phase, even before any lines of code are written. Its purpose is to be aware of potential threats before the actual development and to act on them during that phase, as later adaptions of the system might become complex and costly. Moreover, applying threat modeling techniques uncovers categories of issues that are often overlooked by other tools. For that purpose, two types of models are created. The first depicts the system being built, while the second illustrates the threats that the system might face. These abstractions should foster knowledge about potential risks and enable us to see analogies and similarities of issues in other applications [1].

In this paper, we aim to apply the threat modeling technique to a chosen system of interest. The goal is to gain actionable insights on the approach in practice for an AI-focused blockchain system, called OpenMined. OpenMined is an open-source project with the vision of providing access to data without the negative consequences that might come from misuse (such as IP theft, discrimination, etc.). The provision of decentralized ownership allows for AI models to be trained on data that they cannot itself explicitly access, thus benefiting from the shared information without the downsides associated with direct data ownership. The concept builds on the principle of "Structured Transparency", a term coined by the researchers behind OpenMined. The concept enables everyone to leverage data for the good but not for the bad (i.e. undesired use) [2]. Especially in areas where highly sensitive data is present (e.g. medical records), questions arise on the complicated trade-off between ensuring high privacy standards for that information and also providing researchers access to the valuable data they need for their work on potential scientific breakthroughs [3]. The system of interest attempts to address the issues associated with undesired data usage, making it a particularly interesting candidate for threat modeling. Moreover, this case study about OpenMined could provide relevant insights into other blockchain systems employing AI features and how threat modeling could be applied in similar scenarios. The question we pose is: "What potential security threats does OpenMined face and how could they be addressed in order to deliver a more secure product?". Additionally, the open-source nature of the project allows us to analyze the inner workings of the system, its structure, and code. As a result of this analysis, we can translate the gained insights into the proper models needed for threat modeling.

The state-of-the-art literature proposes multiple potential threat modeling techniques. For the analysis of OpenMined, we chose to use the Hybrid Threat Modeling Method (hTMM). Key features of hTMM include consistent results on repetition and a prioritization of threat mitigation. To achieve these results, hTMM employs a combination of other approaches such as the Security Quality Requirements Engineering method (SQUARE), Security Cards, and Persona non Grata. This should allow us to build a complete threat model, while also eliminating false positives. A more detailed description of the system under evaluation and the methods employed for threat modeling will be provided in the respective sections [4].

II. Background & Related Work

A. Blockchain

Blockchain is referred to as decentralized data management technology with the aim to provide security, data integrity, and anonymity. These benefits manifest themselves due to the absence of a third party that is in control of the transactions and the data. Indeed, it is a decentralized system where every transaction is shared and available to all nodes but without disclosing the identity of the originator [5]. One of the major challenges that have been outlaid by Swan relates to security. Thus blockchains have a possibility of a 51 percent attack where one single entity can get full control over most of the network when it holds more than 51 percent of the computing power [6]. Next, there are fork problems relating to software upgrading. Since blockchains are growing, the scale might also become an issue as the system could become too large for clients to run it. Finally, the decentralized system might weaken the ability to control the data security measures that need to be in place to ensure that highly sensitive data, which might be analyzed in the case of OpenMined, cannot be used for harmful purposes [7].

To overcome security and privacy issues such as transaction linkability, private-key management and recovery, on-chain data privacy, or compliance with regulations such as GDPR, new privacy-preserving techniques are evolving to overcome these challenges. To enable privacy preservation, different approaches have evolved in the last years such as Secure multi-party computation (SMPC), Zero-Knowledge Proofs (ZKP), Commitment schemes, zkSNARK, Ring Signatures, and Homomorphic hiding. A quick overview of the concepts is given below:

1. *SMPC:* divides data or program states between any number of parties using secret sharing [8].
2. *ZKP:* a cryptographic protocol that allows verifying a statement as true without revealing its identity, i.e. it only transfers the proof [9].
3. *Commitment Schemes:* cryptographic tool which allows one party to commit to a secret value without showing it, i.e., if it reveals the original value to another party it will know whether it is true or not [10].
4. *zkSNARK:* a type of highly secure cryptographic testing that builds on ZKP principles and does not require interactions between the prover and the verifier [11].
5. *Ring Signatures:* type of digital signature which does not reveal the person of the group who signed it (for a group with private and public keys) [12].
6. *Homomorphic hiding:* encryption method which allows performing some function over a ciphertext but obtaining the same encrypted result [13].

B. Sensitive Data

The system of interest we chose for threat modeling aims to make sensitive data available for analysis while taking into consideration privacy and security issues that come alongside it. So far we provided a short insight into how the blockchain system is used to benefit the desired approach and some of the privacy and security-related techniques that have been proposed to protect an individual's data. Next, a discussion of the benefits and risks of opening up data for analysis will be conducted, which will be leveraged in the following sections to identify potential threats. Many governments and institutions have started making their data sets publicly available. This step is often justified by the argument that open data will favor innovation through data analysis which, in turn, is set to deliver benefits to the general public. Generally, three major benefits can be associated with open data. First, the effectiveness and efficiency of services can be improved. For example, weather data could be analyzed for the purpose of farming process improvements or health data insights could lead to better treatment of patients. Thus, the quality of services will be improved which will create value for the public. With data being publicly available, this might also increase trust in governmental or institutional decisions and, in turn, their transparency and accountability. Finally, the new data can lead to innovations in other fields and industries, not just foster improvements in the services currently provided [14]. For these reasons, data sharing is highly beneficial, however, it is important not to forget the accompanying challenges and risks when data is shared. Looking at particularly sensitive areas like healthcare, any access to data needs to preserve security and ensure the patients' right to privacy. Furthermore, such systems need to make sure that the data is protected from pernicious attacks and cannot be misused or stolen [15].

C. Threat Modeling

The challenges related to sensitive data make OpenMined an interesting candidate for threat modeling. Threat modelers build two models, one describing the situation and the other of what can go wrong (i.e. the threats). With these two models, one aims to abstract the details of the system to facilitate the goal of thinking about risks. The reason why every system of interest should be subject to some sort of threat modeling is that it allows one to find potential issues early in development. As commonly known, later found bugs in the code or deficiencies with the design of the system are often more expensive and time-consuming to remedy. Next, threat modeling allows one to better understand the security requirements of a system, which facilitates the establishment of requirements for one's system. This leads, in turn, to the ability to provide better and more secure systems. Indeed, by considering the requirements as well as potential threats and vulnerabilities early in the development process, one can think of potential mitigations before building the system. Lastly, it will shed light on issues that otherwise would not be addressed. The models in use allow to abstract away the details and provide an overview of the system which could also be helpful in the design phase [1]. The following is a short overview of common and well-adopted threat modeling approaches, concluding with why we chose the hybrid threat modeling approach.
There exist numerous threat modeling approaches, of which the most known and widely used one is referred to as STRIDE. First, the system has to be modeled with all its data flows, then the model is evaluated based on the different types of threats. STRIDE distinguishes six threat categories: spoofing identity, tampering with data, repudiation, information disclosure, denial of service, and elevation of privilege, towards which the system of interest is checked against [16]. PASTA (Process for Attack Simulation and Threat Analysis) is another well-known technique for threat modeling which consists of

a seven-stage process that determines the effects of threats on applications and the business as a whole [17]. Visual, Agile, and Simple Threat Modeling (VAST) was developed as a basis for commercially available threat modeling techniques. The tool should allow for consistent output and is implemented in the software development life-cycle. It incorporates agile methodologies and is suited for large or medium organizations [18]. Security Cards are developed with the goal of creating a security mindset. Indeed, they should facilitate the exploration of security threats for the system of interest with cards that help to brainstorm potential cyber attacks [19]. Lastly, Persona non Grata represents users behaving in an unwanted, threatening way towards the system. This approach forces the developer to focus more on the attacker, his/her motivations, and goals [20]. Mead and Shull wanted to find an efficient threat modeling method and found that no one solution included all of the identified threats. For this reason, they came up with a new approach called Hybrid Threat Modeling Method (hTMM) which combines some of the presented approaches and thus makes use of the strengths of each [21]. In order to get a significant insight into the threats of OpenMined, we chose the hTMM approach. In a later section, a detailed description of the techniques as well as how we apply hTMM and analyze the system of interest will be given.

D. *Federated Learning*

OpenMined [2] combines several techniques to achieve its goal of using data for data science and machine learning purposes without users having to own or access that data. One of these is the machine learning technique *Federated Learning* [22], [23]. Federated Learning denotes a distributed approach where models are trained on decentralized data residing on remote devices or servers, such as mobile phones. Instead of having to bring the data to the machine that is training the model, the code for training the model is sent to the remote devices, which themselves train the model [24]. The remote devices receive the current model and subsequently train that model with the data of the remote device. Afterwards, the changes made to the model are summarized and sent back, with the training data of the device remaining locally throughout the entire process. Many implementations of Federated Learning also average the updates of several remote devices and do not store individual updates, thereby improving both model quality and privacy [22], [24].

The concrete implementation of Federated Learning used by OpenMined is depicted in figure 1.

First, the data scientist creates the necessary training scripts, models, and configurations, using OpenMined's PySyft library [25]. Second, the created artifacts are hosted on PyGrid [26], which acts as a central server for Federated Learning. Third, the Federated Learning libraries on the remote devices train the model and report the results to PyGrid, where to model is updated. This training and reporting step is repeated according to the number specified in the configuration. As a final step, the trained model is sent back to the data scientist.

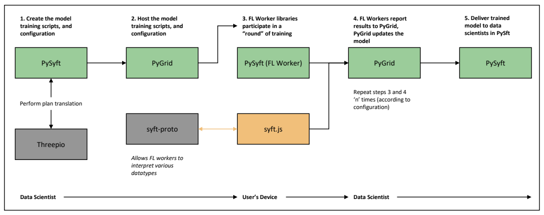
1. Create the model training scripts, and configuration
2. Host the model training scripts, and configuration
3. FL Worker libraries participate in a "round" of training
4. FL Workers report results to PyGrid, PyGrid updates the model
5. Deliver trained model to data scientists in PySft
PySyft
PyGrid
PySyft (FL Worker)
PyGrid
PySyft
Perform plan translation
Threepio
syft-proto
syft.js
Allows FL workers to interpret various datatypes
Repeat steps 3 and 4 'n' times (according to configuration)
Data Scientist
User's Device
Data Scientist

Fig. 1. Implementation of Federated Learning by OpenMined (adapted from [25])

III. Research Method & Methodology

A. *The Hybrid Threat Modeling Method (hTMM)*

For the purpose of their 2018 paper, Mead et. al [4] analyzed some of the most common threat modeling approaches: STRIDE, Persona non Grata, and Security Cards. Using two scenarios (a drone swarm scenario and an aircraft maintenance scenario), the authors attempted to use these methods to identify potential threats. However, none of the approaches was successful in finding all potential threats in both scenarios. As a result, the authors then proceeded to develop hTMM, a method combining these approaches in a more holistic model. When creating the model, a list of desirable characteristics was generated. First, the model should not output any false positives in the analysis. Second, no threats should remain hidden or overlooked by hTMM. Further, the results of the modeling process were supposed to be consistent and independent of the person conducting it. Moreover, using the model should be cost-effective, in terms of time and resources, compared to other models. Finally, the authors sought to provide evidence through empirical studies, supporting the validity of the model. Besides the desirable characteristics mentioned above, other traits were considered when designing hTMM. These were mostly 'softer' criteria and were concerned with the usability and accessibility of the model. One of these traits was, for instance, the desire for the model to be easy to learn and intuitive to use. Another characteristic was for hTMM to support and be compatible with several other tools in the threat modeling sphere.
Generally, hTMM is composed of five distinct steps: First, the system to be threat modeled is identified. For that purpose SQUARE [27] is used. Security Quality Requirements Engineering is a model to examine security/quality requirements early in the product life-cycle, i.e., in the development stage. It is an instrument to elicit, categorize and prioritize security requirements for information technology systems. The first three steps of SQUARE are conducted in this stage of hTMM. First, all participants agree on the distinct definitions used for the system. Then, the business goals, as well as asset and security goals, are identified and defined, while attempting to

collect as many related artifacts as possible.

Second, the Security Cards [4], [19] method is applied. Security Cards is an approach akin to a checklist to help users identify unusual attacks at any stage of the system design process. The card deck consists of 42 cards which are divided into four dimensions. The first one is referred to as Human Impact and deals with the effect on human lives, for example, financial loss. Next, Adversary's Motivations consider the reasons why someone might want to attack the system, e.g., revenge. Adversary's Resources address the means of how an attack would be carried out, e.g., which tools. Lastly, Adversary's Methods shed light on the means through which an attacker might do harm to a system, e.g., through manipulation. To use the method, the Security Cards are distributed to, ideally, all relevant stakeholders. In particular, at least one representative of the system's users, the cybersecurity team, and the engineering team should be present. The participants then read their cards along the four dimensions (human impact and the adversary's motivations, resources, and methods). In the following brainstorming session, each dimension is individually considered and all cards are sorted by the amount of risk they bear. At the end of this step, a list of potentially impacted systems, Personae non Gratae, and potential approaches an attacker might take is compiled.

In the third step, the previously collected results are leveraged to identify and remove the Personae non Gratae which are unlikely to threaten the system. Then, the remaining misuse cases are itemized, meaning the exact attack pattern on the system is described.

After that, a summary of the previous parts is provided in the following step. The identified results are then grouped as follows: attacker, purpose and target of action, description of action, result of action, impact of action, and threat type (per STRIDE element). In the final step, the modelers leverage the summarized results of the previous parts to continue with a formal risk assessment method, e.g. SQUARE.

B. Why hTMM

While many threat modeling methods such as STRIDE, PASTA, VAST Modeling, or OCTAVE exist, we decided upon using the hybrid threat modeling method (hTMM) [28]. The reason for this lies within the method's desirable characteristics which it was built upon, and OpenMined's (further referred to as 'the system') properties, goals, and associated circumstances [4]. One of hTMM's main advantages over other techniques lies within its explicit aim to not overlook any threats. This works well with the system, as OpenMined is built on the key concept of being able to use data owned by other people without the possibility of directly reading said data [2]. Therefore, it is of utmost importance that every possible way of reading, otherwise prohibited, data will be uncovered during the threat modeling process. This threat is critical to the system, as it undermines the central feature of OpenMined. Additionally, the data that is processed by OpenMined is inter alia highly sensitive data which due to their nature requires special care. In this context, other threat modeling methods that tend to overlook some threats such as STRIDE were deemed insufficient for the purpose of analyzing OpenMined. This flaw of STRIDE and other threat modeling methods were proposed by Mead et. al (2018) [4] and is directly addressed by hTMM.

Another reason to use hTMM over other methods lies within OpenMined's open source approach [2]. With the system's source code being available publicly, multiple factors need to be considered when deciding upon which threat modeling method to use. The first set of considerations revolves around multiple people interacting with OpenMined's source code. One can imagine that a number of individual stakeholders are interested in evaluating how secure OpenMined is. Therefore, hTMM's property of delivering consistent results independent of who carries out the threat modeling becomes a desirable characteristic when analyzing an open source system.

Furthermore, it is even more important not to overlook any threats when dealing with an open source system for two additional reasons. First, potential attackers with access to OpenMined's source code are able to detect vulnerabilities they might not detect without access to the source code, giving them a concrete advantage and increasing the danger of potential harm to the system. Second, potential users or partners of OpenMined might analyze the system in order to decide if they want to use it or become partners of the organization. Therefore, not uncovering threats that can be uncovered by everybody through the systems source code must be avoided if possible.

By the same token, one can also assume that critical users want to compare their analysis with others, even when they have no or little prior experience with threat modeling. This can also happen when individuals outside of the field of IT Security but highly interested in the system, such as people working in the healthcare industry, are analyzing it for potential threats. Thus, hTMM's characteristics of delivering consistent results, as well as its intuitiveness, become favorable over other threat modeling methods. Another characteristic of hTMM is the involvement of a variety of different stakeholders via Security Cards. By involving different stakeholders and perspectives, potential threats can be uncovered holistically. This becomes even more beneficial when considering the wide range of different types of data that will be stored on OpenMined, and the different impacts potential insecurities can result in. A further consideration lies within hTMM's threat mitigation prioritization aspect. As mentioned before, OpenMined's key concept is the ability to answer questions using other people's data without directly accessing it. Therefore, preventing unauthorized access to data is definitely one of the highest prioritized threats while other threats might be assigned a lower priority. hTMM allows for such a prioritization by different levels of importance.

Lastly, the method is a rather cost-effective way of threat modeling, according to its authors [4]. As OpenMined is an open source community project and is not a project of a large corporation, it is important to keep the resources used for the threat modeling within reason. Additionally, as other individuals might want to conduct threat modeling on OpenMined themselves and compare their results to others, using a cost-effective method not only benefits the OpenMined community directly but also potential customers and partners of the organization.

C. Evaluation Criteria

In order to evaluate the effectiveness of the chosen method (hTMM) we first need to establish the criteria on which to evaluate it.

Following Shostack's suggestions, one can group them into the categories completeness and correctness [1]. Dealing with every possible threat, for example, is a concern of completeness whereas dealing with threats in the right way is a concern of correctness. Furthermore, Williams and Yuan put emphasis on a threat modeling tool's usability and easiness to understand when evaluating the effectiveness of Microsoft's Threat Modeling Tool by comparing it to a manual threat modeling approach [29]. We categorized these concerns under the title Ease Of Use. These 3 categories do not only align with the reasons why we decided upon using hTMM over other threat modeling methods mentioned in section III.b, but also with 3 of the 5 categories Wuyits et al. used to evaluate the privacy threat analysis methodology LINDDUN. The other 2 categories were productivity, which measured how many valid threats are identified in a given time frame, and reliability, which compared the results of LINDDUN to those of privacy experts [30]. We decided to also include the category productivity in our evaluation while not including reliability. The reason for this is that there are no other cases of threat modeling on OpenMined publicly available at the time of writing.

Therefore, we end up with the following 4 evaluation criteria: completeness, correctness, ease of use, and productivity. We evaluate using these 4 criteria from the perspective of the threat modeling of OpenMined, and from the perspective of hTMM, the threat modeling method. Based on Shostack's suggestions, we will control for completeness for the discovered threats [1]. Generally speaking when talking about completeness with regards to threats, each discovered threat should be documented and labeled as a bug and it should be notified if each bug has been dealt with or not. Bugs can be categorized as not important enough to be addressed directly but to be monitored instead. In this case, it is again important to ensure that this monitoring is indeed happening and should not be forgotten. Bugs that are unaddressed should gain priority the longer they stay unaddressed.

The threat's correctness should be evaluated by ensuring that these threats in fact exist and are dealt with correctly. This would be done by implementing test cases, i.e. trying to execute said threat without and with protection mechanisms in place. These test cases can be manual or automated in nature. Shostack further suggests that each solution to a threat in place should be further tried to be bypassed in order to ensure its effectiveness [1]. As we are not part of the OpenMined development team and because it would exceed the scope of the paper, we will not implement test cases or solutions ourselves. However, we will provide a valuable set of questions that should be considered by the OpenMined community when doing so.

1) Is a bug report realistically writable for this threat?
2) How easily can a solution for this threat be conceived?
3) Are there potential explicit solutions for this threat that are standing out?
4) How easy and cheap would it be to implement a solution to this threat?
5) Can the system be tested for this threat?
6) Are potential tests for this threat difficult to create?
7) Can this threat be tested manually, automatically, or both?

Productivity will be measured by computing how many valid threats have been identified by a participant within the time spent on threat modeling.

Last, hTMM will be evaluated for its ease of use by conducting an open survey among the participants alongside the feedback categories similar to those found within Williams and Yuan's study on evaluating the effectiveness of Microsoft's Threat Modeling Tool. These categories are the challenges participants had with the method, features and concepts that participants liked about the method, and participants suggestions for improving the method [29].

IV. Results

A. Definitions

In adherence with the first three steps of SQUARE, we first have to agree on the definitions for the terms used. Here, as suggested by Mead et al. [27], we will use the latest version of the ISO/IEC/IEEE Standard Vocabulary of Systems and software engineering [31]. Therefore, we will define data as a "representation of facts, concepts, or instructions in a manner suitable for communication, interpretation, or processing by humans or by automatic means" [31]. Further, we will define data use as an "executable statement where the value of a variable is accessed" and data provider as an "individual or organization that is a source of data" [31]. This allows for a separation between the entity which uses the data (data user) and the entity which provides the data (data provider). Additionally, data protection will be understood as the "implementation of appropriate administrative, technical, or physical means to guard against unauthorized intentional or accidental disclosure, modification, or destruction of data", while we will define authorization as an "action indicating that a particular behavior shall not be prevented" [31]. These definitions are beneficial for a proper identification of the central business and system goals of OpenMined.

In [3], some of the individuals behind OpenMined, further provide a useful taxonomy for the purpose of this paper. The authors detail the privacy trade-off, which can be described using the prior definitions by the following: one could either benefit from data use or the retention of data protection, but not both. By the same token, the authors argue that the aim of overcoming this trade-off can be called ”structured transparency” [3], which consists of input privacy and output privacy. Output privacy can be defined as ”preventing the output of the flow from being reverse engineered to reveal additional information”, while input privacy is concerned with ”facilitating the flow of information without leaking collateral data” [3]. Further, the authors discuss, data needs verification in order for the recipients to trust it. They distinguish between input verification, which signals the recipient that the data originates from a trusted source, and output verification, which allows to verify if the data was modified. Flow governance, in addition, is given when all parties are guaranteed that the intended use of the message flow will remain the same.

The authors then describe three distinct limitations of structured transparency which occur in practice [3]. The most common is the copy problem, which describes the unrestricted ability of the recipient to further share the data. The bundling problem is concerned with the issue of needing to reveal additional information when sharing a specific bit of information, because it is either needed when decoding or verifying the authenticity of the particular bit of information. While oversight sometimes solves both issues, it can cause the ”recursive oversight problem”: the institution in charge of the oversight, in turn, requires oversight [3].

B. Business and Security Goals

Turning towards the business goals, as in the second step of SQUARE, OpenMined aims to enable individuals ”to answer their most important questions by empowering them to learn from data owned and governed by others” [2]. Using our definitions from above, we can partly reformulate this mission into: help individuals engage in authorized data use without necessarily being the data provider. Therefore, one of the key security goals essential to OpenMined is data protection, especially in the context of unauthorized data use. The central security challenge here thus can be described as allowing for authorized data use while preventing unauthorized data use, which is complicated by the fact that the data user is often not the data provider [2]. Using Trask et al.’s (2020) definition [3], we can add that OpenMined attempts to evade the privacy trade-off by (partly) achieving structured transparency. As discussed above, this means attaining both output and input privacy. Additionally, as described before, it requires for input and output verification, as well as flow governance. In doing so, OpenMined must navigate the three challenges of structured transparency: the copy problem, the bundling problem, and the recursive oversight problem. The security goals are in clear support of the organization’s business goal.

OpenMined is not only a community project, but also a charity [2]. That means that the business goal is not profit driven or monetary, but altruistic. The organization attempts to achieve its goals using three distinct programs. The first and most central is the open source software build by the OpenMined community. The software solution has two aspects. The first exists for the purpose of providing data for others to study. It is realized by being uploaded on a PyGrid server, which is designed for individuals to be able to use the data, but not download it [26]. On the other side, PySift is the library used to perform data science on someone else’s data available through a PyGrid server [26]. Further, OpenMined provides an educational program, which trains individuals on using PySift and PyGrid, as well as on general topics around privacy and data. Lastly, the organizations partners with the private and public sector to enable them to use their software in novel use cases [2].

C. Artifacts

Below a short list of relevant artifacts, as per the third step of SQUARE [27].

- Information Security and Identity Team Workplan and Roadmap [32]
- Vulnerability Analysis for Vulnerability A (reverse engineering, approximating, or reversing values or labels) [33]
- Vulnerability Analysis for Vulnerability B (approximating values) [34]
- COVID Use Case [35]
- OpenMined Security and Identity Team [36]
- PySift Issue Tracker [37]
- OpenMined Team Structure [38]
- OpenMined Use Cases [39]
- OpenMined Products [40]

D. Security Cards

As a second method, hTMM employs Security Cards to encourage threat modelers to to think about a broad range of potential security threats [19]. In total there are 42 cards in four categories covering the methods, motivations and resources of adversaries as well as the human impacts of the discovered threats. The following section presents the results of applying Security Cards to OpenMined, divided into the four categories.

1) Adversary’s Methods

1. Attack Cover-up relates to how the awareness of an attack can be altered. In the case of OpenMined there is a family of attacks around hosting misleading data disguised as legit data to worsen model performance. This type of attack can only be detected if the model performance can be assessed as there is no access to the data itself, and therefore no way to directly verify the data.
2. Indirect Attack relates to how unexpected or secondary system properties could be used to attack the system. Statistics about the popularity of data sets could be used to spoof popular data sets and host nonsensical data sets with similar names/properties.
3. Manipulation or Coercion relates to how people with access to the system could be persuaded to perform actions that are detrimental to the security of the system. Trusted institutions/individuals could be blackmailed into uploading bad data under their name.

4. Multi-Phase Attack relates to how multiple attacks can be combined to execute a more effective attack. If a machine learning model is not processed independently of parties with prior knowledge, an attacker could reverse engineer data values by gaining access to both the weights, and biases of the model as well as the output produced by the model [33]. This attack pattern is especially relevant if this information has already been published, and is therefore easily available to potential attackers. The multi-phase attack would then include as a first step gaining access to the required information and as a second step, to reverse engineer the underlying data, undermining OpenMined's goal of keeping the data private and secure.
5. Physical Attack relates to how physical access to a system can be leveraged for an attack. The servers of OpenMined, or an institution hosting data through OpenMined, could be attacked by tampering with the respective soft- or hardware, e.g., installing keyloggers, disconnecting drives.
6. Processes relates to how organizational or technical processes can be exploited for an attack. In the case of OpenMined the process governing how institutions/individuals identify themselves could be exploited to impersonate reputable data sources (e.g., universities) and therefore needs to be resilient against such attacks. Further, the omission of a process assessing the quality of hosted data opens the door for attackers to host nonsensical or malicious data. Additional process attacks include those that take advantage of vulnerabilities within the methods employed by OpenMined to achieve the desired properties, i.e. allowing to train models on not directly accessible data. When using federated learning and split neural networks, it is crucial that the model is processed independently of the data owner, or other parties with prior knowledge of the model or underlying data, otherwise it is possible to reverse engineer [33] or approximate [34] values of the underlying data.
7. Technological Attack relates to how technical attacks can be performed via digital or analog links. Some of the possible attacks have been described in relation to prior security cards (e.g., spoofing reputable data hosts). Further attacks include (distributed) denial of service attacks against data hosts or OpenMined.
8. Unusual Methods refer to any other methods that can be used for an attack. In the case of hosted data it needs to be ensured that the integrity of these data sets is ensured and data users are made aware of any (incidental or malicious) changes, otherwise altering data sets could deliver different results over time.

2) Adversary's Motivations

1. Access or Convenience refers to how a system can be misused out of convenience, or for accessing some resource. As OpenMined hinders direct data access, it is an easy tool to conceal poor data quality. Further, it can also serve as a convenient way to claim that some data set is someone's own, when it is actually taken from another source.
2. Curiosity or Boredom refers to attacks carried out in relation to curiosity or out of boredom. Trying to access the original data via approximation or reverse engineering could be motivated by curiosity, while uploading poor data to worsen someone else's models could be out of boredom.
3. Desire or Obsession refers to individuals that attack a system out of obsession with the system or some related individual. We deemed that this security card is irrelevant for OpenMined.
4. Diplomacy or Warfare refers to a system being attacked to gain diplomatic or military advantages. We deemed this card irrelevant as OpenMined has not reached the importance or influence needed to gather the attention of state actors.
5. Malice or Revenge refers to how a system can be attacked for reasons of malice or revenge. Disgruntled ex-employees could harm their ex-employers reputation by hosting poor data sets under the employers name. Further attacks out of malice include DDoS attacks or the reverse engineering and subsequent publishing of data sets with the intent to cause reputational or monetary damage. *6. Money* refers to attacks on a system out of a financial motive. By getting access to the underlying data via reverse engineering or approximation, attackers could sell the obtained data sets. Performing a DDoS attack could be used to extort money from the affected institutions.
7. Politics refers to attacks that could be used for political gains. Politically useful misinformation that is backed by data and comes from a credible institution could be used to spread political opinions and to further the attacker's agenda. As the data is not accessible and verifiable by others, an attacker could generate data that fit their preconceived results. Further, the data host could be impersonated to seem (more) credible.
8. Protection refers to how a system can be utilized to protect oneself or others. We deemed this security card irrelevant for OpenMined.
9. Religion refers to attacks out of a religious motive. This card was also deemed irrelevant in the context of OpenMined.
10. Self-Promotion refers to how an attacker could misuse a system to promote themselves. An adversary could generate fake uses of their data to appear as a popular data host or could host popular data sets from other sources as their own to boost their credibility.
11. Unusual Motivations were deemed irrelevant in the context of this threat modeling project.
12. World View refers to how a system can be misused to further the attacker's worldview. The attacks and methods are similar to those specified for the politics security card.

3) Adversary's Resources

1. Expertise refers to the skill levels of an attacker. Data science experts could upload poor data that is not easily identified as such or could use their skills and knowledge to reverse engineer or approximate the original data.
2. A Future World refers to capabilities that could be available to attackers in the future. Increased usage and popularity

of OpenMined, potentially fueled by new data or privacy regulations, render it a more valuable target.
3. *Impunity* refers to the possibilities of attackers to get away with their actions. Anonymity and/or living in rogue states which do not prosecute or even support cyber attacks lower the barrier for all sort of attacks.
4. *Inside Capabilities* refers to capabilities that are acquired by having inside access to the system or an inside source. As OpenMined is designed as an open-source project, inside capabilities could be acquired by being part of the core project team and having expanded capabilities, e.g., approving fraudulent/malicious pull requests.
5. *Inside Knowledge* refers to knowledge that is not publicly available and can be used to facilitate an attack. At first glance, inside knowledge seems to be a less of a problem in an open-source environment. Yet, there are still areas that have to be considered as the source code might be public but not every information about the system is. For example, there could be vulnerabilities that are not public and could therefore used by insiders as an attack vector.
6. *Money* refers to an adversary's financial capabilities. Financially well-situated attackers would be able to bribe OpenMined contributors or credible data hosts to conduct actions in the attacker's interest. Moreover, such adversaries would also have access to more sophisticated tools, e.g., larger botnets for DDoS attacks.
7. *Power or Influence* refers to the power or influence of an attacker which can be leveraged to attack a system. apart from the financial capabilities of an attacker, which have been covered by the previous security card.
8. *Time* refers to the time limits an attacker faces when carrying out an attack. The different attacks presented in the motivations and methods section target different timelines. DDoS attacks tend to be a short-term attack, while reverse engineering data would require a considerably higher time effort to gain access to the required information to actually perform the reverse engineering. The timelines for impersonating data hosts and the uploading of poor or stolen data sets depends on the tools employed by OpenMined to detect these types of misuse and to hinder further attacks by the same adversary. It has to be said however, that both type of attacks are rather quickly to conduct.
9. *Tools* refers to the tools necessary to carry out an attack. The tools needed depend heavily on the type of attack an adversary wants to conduct. DDoS attacks presuppose having access to a botnet, while reverse engineering and approximation efforts require some sort of prior knowledge or access to model parameters. For hosting poor or stolen data sets, a webserver (or a PC that can act as one) is needed. The same applies for impersonating a credible data host.
10. *Unusual resources* refer to uncommon resources an adversary might have access to. They were deemed irrelevant or covered by other security cards in the present context.

4) Human Impact

1. *Emotional Wellbeing* refers to a system's impact on the emotional wellbeing of individuals. This card was deemed as not directly relevant in the context of OpenMined, although there could be implications to third parties if the misuse of personal data of them leads to emotional harm. For example, if medical records hosted by a medical institution the third party visited as a patient could be reverse engineered.
2. *Financial Wellbeing* refers to a system's impact on individual's financial assets. In the present context, this card was deemed irrelevant.
3. *Personal Data* refers to personal data a system uses and the implications of misuse of personal data. The properties and features offered by OpenMined present it as an enticing solution for data science tasks on sensitive data. However, this also leads to the requirement to safeguard personal data against any type of misuse. As has been discussed previously, there are ways to gain access to the underlying data, either through reverse engineering or approximation. If an attacker has or obtains access to information needed to reverse engineer/approximate data, which is perfectly possible, personal data could come under attack.
4. *Physical Wellbeing* refers to a system's impact on individual's physical wellbeing. This security card was deemed irrelevant in the present context.
5. *Relationships* refers to the impacts a system can have on relationships. A company's reputation or that of the responsible employee(s) could be damaged if the data hosted by them can be reverse engineered, or approximated and subsequently misused by the attacker.
6. *Societal Wellbeing* refers to a system's impacts on the financial, physical and emotional wellbeing of a society. Given the size of OpenMined, even a large scale attack would not have societal impacts as defined by the security card. Therefore, this card was deemed irrelevant.
7. *The Biosphere* refers to a system's impact on the environment. In the context of OpenMined, this card was deemed irrelevant.
8. *Unusual Impacts* refer to special or exceptional values that might be impacted by a system. In the present context, trust plays an important role between multiple actors. Data hosts need to be able to trust OpenMined's methods to keep the hosted data private, while at the same time data users need to be able to trust the hosted data, meaning they perform data science on credible data. By extension, data users must trust data hosts and vice versa. If faith is missing, neither data hosts, nor data users will find OpenMined a viable platform for privacy-focused AI and data science. Each attack or even possibility of attack is certainly detrimental to establishing trust between the involved parties. In theory, an attack on OpenMined could also influence the trust in decentralized (blockchain) applications, yet this seems highly unlikely. This is because, unlike large blockchain systems like Bitcoin or Ethereum, OpenMined has not reached the size needed to have this level of impact.

E. Persona non Grata

Another element of the hTMM method is the Persona non Grata (PnG) analysis [4]. Thinking about different archetypes of potential attackers in the process of the PnG analysis resulted in the following potential attackers:

1) The Troll: As a high-school student during the summer, Dominik has a lot of free time on his hand. The lack of things to do combined with the fact that most of his friends are on vacation left Dominik bored and spending most of his time on the internet.
Being bored on the internet, Dominik started annoying people online to entertain himself.
Dominik's misuse cases are:

1) Uploading bad or unusable data under a promising name to deliver bad model results for data users.
2) Changing already uploaded data after some time to destroy the validity of models that use the same data set at a later time.
3) Upload fraudulent or illegal data which is unsuitable for sharing.

As a troll, Dominik's single goal is to mess with people for his own entertainment.
Dominik has enough time on his hands to teach himself to use PyGrid and PySift, which are free and explained with OpenMined Tutorial videos.

2) Disgruntled Ex-Employee: Sandra used to work as a researcher at a credible university. As a researcher she used to upload data sets for her university to OpenMined. Despite her being very passionate about her field of study she got fired due to her continuously violating the university's regulations and her bad attitude towards students.
Sandra's misuse cases are:

1) Uploading bad or unusable data under the university's name.
2) Leak sensitive data or giving access to it (harming input transparency)
3) Use her knowledge about the university's inner workings, her relationships or existing access to the university's systems to tamper with the data uploaded by the university or do any of the above.

Sandra's goal is to hurt her former employer's reputation and harm it financially. While Sandra does not has any technical expertise she is very knowledgeable about the university's inner workings and the data she uploaded to OpenMined for her university.

3) Presumptuous Hacker: Markus is a skilled hacker who loves to proof his abilities and to challenge himself. He is very knowledgeable in the fields of data science and artificial intelligence. Markus also developed a particular interested in blockchain technology over the past few years, constantly looking for new ideas and systems in this field.
Markus' misuse cases are:

1) Reverse engineering uploaded data by statistically approximating its results , gaining indirect access to private data (harming output privacy).
2) Accessing restricted data from PyGrid servers to show a lack of security mechanisms covering such cases.
3) Start a denial of service attack on OpenMined to show that it cannot provide an adequate service level.

Markus' goal consists of showing off his hacking skills and therefore increasing his reputation as a hacker.
Markus' skills consist of strong coding and hacking abilities as well as profound knowledge in the fields of data science, artificial intelligence, and blockchain technologies.

4) (State-sponsored) Activist Groups: CloseMinded is a radical activist group which is responsible for disrupting events and attacking ideological enemies, such as members of certain political parties or celebrities that are vocal about certain topics. Therefore, the group receives (financial) support from sometimes very powerful radical individuals.
CloseMinded's misuse cases are:

1) Starting denial of service attacks on its enemy institutions' host providers.
2) Spoofing data set origins, pretending to be an organization they are not.
3) Compromising data from PyGrid Servers of opposing organizations.

CloseMinded's goals are to extort money from certain institutions as well as to harm enemy institutions both financially and reputation-wise. Due to its strong support and its considerable size CloseMinded's resources include money, technical know-how among some of its members as well as vast computational resources.
Looking at these Persona non Grata's misuse cases in combination with the data users and data providers use cases, the system's resulting relevant security cases consider ensuring access to the data and the service, ensuring confidentiality of the data provided, ensuring integrity of the data provided, as well as ensuring authenticity of the data provide. This approach of designing for different archetypical users and relating their misuse cases to normal users use cases was based on Hueng Cleland-Huang's (2014) work [20]. Figure 2 summarizes this relationship.

F. Summary of Results

Table 1 shows a selection of some of the central threats we identified. The actors for each host are taken from the Personae non Gratae named in the previous section: (1) The Troll, (2) Disgruntled Ex-Employee, (3) Presumptuous Hacker, (4) (State-sponsored) Activist Group. Each of these threats can be associated with one of three higher motivations. The first group is concerned with discrediting the data hosting organization, OpenMined, or both. Especially impersonating the data host, a form of spoofing, can have strongly negative impacts on the reputation of all participants and potentially endanger the entire project. Limiting access to the service

TABLE I
Summary of Identified Threats

Threat Name	Actor (PnG)	Purpose	Target	Impact	Threat Type
Hosting Stolen/Illegal Data	1	Discrediting	Host	Medium	Info. Disclosure
Impersonate Data Host	1, 2, 4	Discrediting	User	Severe	Spoofing
Host Poor Data	1, 2	Discrediting	User	Medium	Tampering
Reverse Engineer/Approximate Data	3,4	Data Theft	Host	Severe	Info. Disclosure
DDoS Data Host	3, 4	Limit Access	Host	Low	Denial of Service
DDoS OpenMined	3, 4	Limit Access	OpenMined	Medium	Denial of Service
Compromise PyGrid Server	2, 3, 4	Data Theft	Host	Severe	Info. Disclosure
Covertly Alter Hosted Data	1, 2	Data Theft	User + Host	Medium	Tampering

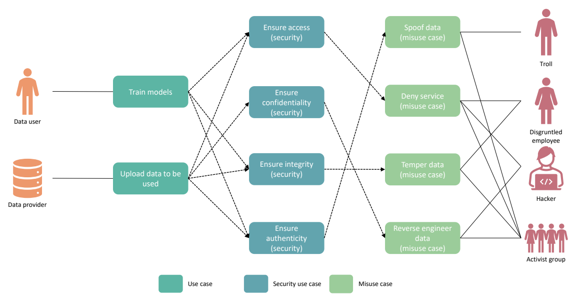

Fig. 2. Misuse cases showing the interaction of malicious users with OpenMined (adapted from [20])

by means of DDoS attacks are also accounted for. Lastly, the attacks with the purpose of data theft have most severe potential consequences. This is because those attacks go against input and output privacy, which are the core goals of OpenMined. Both reverse engineering/approximating data and compromising PyGrid servers could severely endanger the entire project and its success. These threats thus should be prioritized.

V. Evaluation

The evaluation covers both the threat modeling of the OpenMined system as well as the applied threat modeling method hTMM. For this, we use the four criteria specified in section III.C to assesses the output of the threat modeling as well as hTMM as a methdod.
The first criterion, completeness, is concerned with whether all threats applicable to a system have been identified. We would rate the completeness of our threat model as medium because of the manageable but not excessive list of identified threats. This can be ascribed directly to the applied method that acts more as a loose guideline than a granular step-by-step manual. Hence, the output of the threat modeling is directly dependent on the experience of the threat modelers. This becomes especially apparent by comparing hTMM to more detailed and structured approaches like attack trees or attack libraries. Although it has to be noted that those methods have their own drawbacks, namely they may not be available/applicable for the system of interest and offer far less freedom when threat modeling.

The second criterion, correctness, is concerned with the factual truth of the identified threats. We would rate the completeness of our threats as high. Here hTMM can exert its full potential by combining SQUARE, Security Cards and Persona non Grata. This combination of different approaches forces the threat modelers to think multiple times about the same (type of) threats, each iteration leading to a more detailed understanding as well as assessing each threat from different perspectives. The latter is especially facilitated by the use of security cards and the different angles contributed by each participant. Correctness is also high because one step of SQUARE is concerned with identifying important artifacts of the system of interest, which can be used for threat elicitation as well as to review and verify each found threat against some artifact.

The third criterion, ease of use, is concerned with the difficulty of the applied method. We found hTMM to be an easy to use method, even for beginners of threat modeling, and therefore would rate its ease of use as high. This can be attributed to the freedom the method provides, yet still giving guidance on what steps need to be performed. Furthermore, the inclusion of artifacts from the system of interest as well as the collaborative approach during the security cards step simplify applying the method, even for non-specialists.

The fourth and final criterion, productivity, is concerned with the effort needed to apply a certain threat modeling method. We would rate the productivity of hTMM as medium. Reasons for this can be found in the lack of a step-by-step guidance and a freer approach to threat modeling, which leads to a higher effort to identify threats. Moreover, this also leads to longer considerations on the exact nature and properties of a threat. As alluded to in the paragraph regarding completeness, finding all applicable threats with hTMM would require considerable effort, especially for inexperienced users.

Discussing the methods among the authors of this paper, who were also responsible for the various threat modeling steps, led to uniformly positive feedback regarding hTMM. The medium ratings for the completeness and productivity criteria can be attributed to the novelty of the task for the authors. Generating a threat model that deserves a high rating for all criteria, requires a high level of familiarity and

knowledge with the task and system at hand, which cannot be expected from first-time threat modelers. However, hTMM forces threat modelers to think about threats multiple times and from a variety of perspectives, leading to a high rating for correctness. In combination with the high ease of use, we recommend hTMM for everyone who thinks about threat modeling, beginners and experts alike.
OpenMined should consider the identified threats and respond with appropriate mitigations to address them. At a bare minimum, (potential) users should be alluded to the threats using the system poses. Accordingly, they can adapt their own behavior, e.g., checking data hosts, only hosting non-personal/non-confidential data, until viable and tested mitigations are in place.

VI. Conclusion & Future Work

Finding threats and security issues is especially relevant for systems dealing with sensitive data, as the misuse of such may lead to negative impacts on individuals and society as a whole [15]. Indeed, as the Personae non Gratae show, there are several types of attackers with different resources that might want to misuse or exploit data in ways that have negative consequences for others. The hTMM approach is said to have the benefit of finding a significant number of severe threats compared to other threat modeling approaches, due to the combination of multiple techniques that evolved in recent years [4].

The paper's aim was to deploy the hTMM approach to the system of interest (OpenMined), in order to check how well the technique can be used and how many and important threats can be found. Several threats and attacker personas with corresponding motivations could be outlined, which the OpenMined system developers should consider in order to make the system more secure. For them, it is recommended to think about the mitigations of these threats, how the mitigations can be bypassed, how the bypasses could be mitigated, and so on [1]. Moreover, having multiple stakeholders involved in the threat modeling will increase the completeness and correctness of the approach.

In order to check the validity of the results and to increase their reproducibility we suggest three measures. First, the same technique (hTMM) should be applied by different researchers to the same system (OpenMined). This would allow to see whether the method used is applied correctly and consistently. Next, using the same technique and applying it to a different system of interest allows to check if the hTMM approach is beneficial in other settings too, or if it is context dependent. Lastly, using a different threat modeling approach on the same system would significantly contribute to validating or rejecting the results found as well as scanning for potential blind spots of hTMM.

Future studies could further examine other open source solutions which make research data available to the general public. Here, the findings of this paper can be compared to similar systems, potentially expanding them. The Personae non Gratae and adversary's methods we discussed could be applied to projects in a similar context. Also, the use of hTMM could be further researched to gather additional evidence for its use cases. Researchers could, however, not only focus on checking for other approaches, or applying hTMM to different systems, but focus on mitigation techniques to combat these threats. Evaluating and prioritizing the threats that have been found here and thinking about their motivations could be an important step to prevent future attacks on such systems from happening.

References

[1] A. Shostack, *Threat modeling: Designing for security*. John Wiley & Sons, 2014.

[2] OpenMined, "Openmined website," 2022. [Online]. Available: https://www.openmined.org

[3] A. Trask, E. Bluemke, B. Garfinkel, C. G. Cuervas-Mons, and A. Dafoe, "Beyond privacy trade-offs with structured transparency," *arXiv preprint arXiv:2012.08347*, 2020.

[4] N. R. Mead, F. Shull, K. Vemuru, and O. Villadsen, "A hybrid threat modeling method," *Carnegie MellonUniversity-Software Engineering Institute-Technical Report-CMU/SEI-2018-TN-002*, 2018.

[5] J. Yli-Huumo, D. Ko, S. Choi, S. Park, and K. Smolander, "Where is current research on blockchain technology?—a systematic review," *PloS one*, vol. 11, no. 10, p. e0163477, 2016.

[6] M. Swan, *Blockchain: Blueprint for a new economy.* " O'Reilly Media, Inc.", 2015.

[7] I.-C. Lin and T.-C. Liao, "A survey of blockchain security issues and challenges." *Int. J. Netw. Secur.*, vol. 19, no. 5, pp. 653–659, 2017.

[8] R. Cramer, I. Damgård, and J. Nielsen, *Secure Multiparty Computation and Secret Sharing*. Cambridge University Press, 2015.

[9] S. Goldwasser, S. Micali, and C. Rackoff, "The knowledge complexity of interactive proof systems," *SIAM Journal on computing*, vol. 18, no. 1, pp. 186–208, 1989.

[10] G. Brassard, D. Chaum, and C. Crépeau, "Minimum disclosure proofs of knowledge," *Journal of computer and system sciences*, vol. 37, no. 2, pp. 156–189, 1988.

[11] B. Parno, J. Howell, C. Gentry, and M. Raykova, "Pinocchio: Nearly practical verifiable computation," in *2013 IEEE Symposium on Security and Privacy*. IEEE, 2013, pp. 238–252.

[12] R. L. Rivest, A. Shamir, and Y. Tauman, "How to leak a secret," in *International conference on the theory and application of cryptology and information security*. Springer, 2001, pp. 552–565.

[13] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proceedings of the forty-first annual ACM symposium on Theory of computing*, 2009, pp. 169–178.

[14] K. Hardy and A. Maurushat, "Opening up government data for big data analysis and public benefit," *Computer law & security review*, vol. 33, no. 1, pp. 30–37, 2017.

[15] K. Abouelmehdi, A. Beni-Hessane, and H. Khaloufi, "Big healthcare data: preserving security and privacy," *Journal of big data*, vol. 5, no. 1, pp. 1–18, 2018.

[16] L. Kohnfelder and P. Garg, "The threats to our products," *Microsoft Interface, Microsoft Corporation*, vol. 33, 1999.

[17] T. UcedaVelez, "Real world threat modeling using the pasta methodology," *OWASP App Sec EU*, 2012.

[18] "Real world threat modeling," May 2022. [Online]. Available: http://threatmodeler.com/

[19] T. Denning, B. Friedman, and T. Kohno, "The security cards: A security threat brainstorming toolkit," *Univ. of Washington, http://securitycards. cs. washington. edu*, 2013.

[20] J. Cleland-Huang, "How well do you know your personae non gratae?" *IEEE software*, vol. 31, no. 4, pp. 28–31, 2014.

[21] N. Mead and F. Shull, "The hybrid threat modeling method," Carnegie Mellon University's Software Engineering Institute Blog, Apr. 23 2018 [Online]. [Online]. Available: http://insights.sei.cmu.edu/blog/the-hybrid-threat-modeling-method/

[22] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečnỳ, S. Mazzocchi, B. McMahan *et al.*, "Towards federated learning at scale: System design," *Proceedings of Machine Learning and Systems*, vol. 1, pp. 374–388, 2019.

[23] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[24] B. McMahan and D. Ramage, "Federated learning: Collaborative machine learning without centralized training data," 2017. [Online]. Available: https://ai.googleblog.com/2017/04/federated-learning-collaborative.html

[25] OpenMined, "Openmined project layout," 2022. [Online]. Available: https://lucid.app/lucidchart/4171bac3-56e3-490a-85cd-cc7c120151c9/edit?shared=truepage=uZIWEWUiN Db

[26] ——, "Openmined github," 2022. [Online]. Available: https://github.com/OpenMined

[27] N. R. Mead and T. Stehney, "Security quality requirements engineering (square) methodology," *ACM SIGSOFT Software Engineering Notes*, vol. 30, no. 4, pp. 1–7, 2005.

[28] N. Shevchenko, T. A. Chick, P. O'Riordan, T. P. Scanlon, and C. Woody, "Threat modeling: a summary of available methods," Carnegie Mellon University Software Engineering Institute Pittsburgh United . . . , Tech. Rep., 2018.

[29] I. Williams and X. Yuan, "Evaluating the effectiveness of microsoft threat modeling tool," in *Proceedings of the 2015 information security curriculum development conference*, 2015, pp. 1–6.

[30] K. Wuyts, R. Scandariato, and W. Joosen, "Empirical evaluation of a privacy-focused threat modeling methodology," *Journal of Systems and Software*, vol. 96, pp. 122–138, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S016412121400137X

[31] IEEE, *24765-2017 - ISO/IEC/IEEE International Standard - Systems and software engineering–Vocabulary*. The Institute of Electrical and Electronics Engineers, 2017.

[32] A. J. Hall, "Information security and identity team workplan and roadmap," 2020. [Online]. Available: https://github.com/OpenMined/Roadmap/blob/master/security_and_identity_team/workmap.pdf

[33] ——, "Splitnn / federated learning vulnerability a," 2020. [Online]. Available: https://github.com/OpenMined/Roadmap/blob/master/security_and_identity_team/documentation/tools/DoIA/SplitNN/VulnA.pdf

[34] ——, "Split neural network vulnerability b," 2020. [Online]. Available: https://github.com/OpenMined/Roadmap/blob/master/security_and_identity_team/documentation/tools/DoIA/SplitNN/VulnB.pdf

[35] ——, "Covid self-sovereign identity," 2020. [Online]. Available: https://github.com/OpenMined/Roadmap/tree/master/security_and_identity_team/projects/COVID-SSI/B

[36] A. Trask and A. J. Hall, "Security and identity team," 2020. [Online]. Available: https://github.com/OpenMined/Roadmap/tree/master/security_and_identity_team

[37] n.a., "Pysift issue tracker," 2022. [Online]. Available: https://github.com/OpenMined/PySyft/issues

[38] A. Trask and S. Das, "Openmined teams," 2021. [Online]. Available: https://github.com/OpenMined/OM-Welcome-Package/blob/master/images/om-teams-diagram-Oct2020.png

[39] OpenMined, "Concepts and use cases," n.a. [Online]. Available: https://lucid.app/lucidchart/4171bac3-56e3-490a-85cd-cc7c120151c9/edit?shared=truepage=uZIWEWUiN Db

[40] ——, "Products," n.a. [Online]. Available: https://lucid.app/lucidchart/4171bac3-56e3-490a-85cd-cc7c120151c9/edit?shared=truepage=.dNWx-JB4p7r