SoK

Explainable Machine Learning for Computer Security Applications

Nadeem, Azqa; Vos, Daniël ; Cao, Clinton; Pajola, Luca; Dieck, Simon; Baumgartner, Robert; Verwer, Sicco

DOI
10.1109/EuroSP57164.2023.00022

Publication date
2023

Document Version
Final published version

Published in
Proceedings of the 2023 IEEE 8th European Symposium on Security and Privacy (EuroS&P)

2023 IEEE 8th European Symposium on Security and Privacy (EuroS&P) | 978-1-6654-6512-0/23/$31.00 ©2023 IEEE | DOI: 10.1109/EuroSP57164.2023.00022

SoK: Explainable Machine Learning for Computer Security Applications

Azqa Nadeem
Delft University of Technology
Delft, The Netherlands
azqa.nadeem@tudelft.nl

Daniël Vos
Delft University of Technology
Delft, The Netherlands
d.a.vos@tudelft.nl

Clinton Cao
Delft University of Technology
Delft, The Netherlands
c.s.cao@tudelft.nl

Luca Pajola
University of Padua
Padua, Italy
pajola@math.unipd.it

Simon Dieck
Delft University of Technology
Delft, The Netherlands
s.dieck@tudelft.nl

Robert Baumgartner
Delft University of Technology
Delft, The Netherlands
r.baumgartner-1@tudelft.nl

Sicco Verwer
Delft University of Technology
Delft, The Netherlands
s.e.verwer@tudelft.nl

***Abstract*—Explainable Artificial Intelligence (XAI) aims to improve the transparency of machine learning (ML) pipelines. We systematize the increasingly growing (but fragmented) microcosm of studies that develop and utilize XAI methods for defensive and offensive cybersecurity tasks. We identify 3 cybersecurity stakeholders, *i.e.,* model users, designers, and adversaries, who utilize XAI for 4 distinct objectives within an ML pipeline, namely 1) XAI-enabled user assistance, 2) XAI-enabled model verification, 3) explanation verification & robustness, and 4) offensive use of explanations. Our analysis of the literature indicates that many of the XAI applications are designed with little understanding of how they might be integrated into analyst workflows – user studies for explanation evaluation are conducted in only 14% of the cases. The security literature sometimes also fails to disentangle the role of the various stakeholders, *e.g.,* by providing explanations to model users and designers while also exposing them to adversaries. Additionally, the role of model designers is particularly minimized in the security literature. To this end, we present an illustrative tutorial for model designers, demonstrating how XAI can help with model verification. We also discuss scenarios where interpretability by design may be a better alternative. The systematization and the tutorial enable us to challenge several assumptions, and present open problems that can help shape the future of XAI research within cybersecurity.**

***Index Terms*—XAI, Machine learning, Cyber security.**

1. Introduction

Security practitioners are interested in high-performing machine learning (ML) systems that can also explain their decisions [111]. However, despite the unprecedented performance achieved by prevailing ML systems, they have been slow to materialize in the security industry [13], [61], [121]. This is because these systems are considered 'black boxes' due to their lack of transparency — they are notoriously difficult to understand for humans because of their complex configurations and large model sizes. In addition to the lack of understandability, their correctness and robustness can also not be easily verified. For instance, the model might learn incorrect associations (*i.e.,* spurious correlations) from the input data, giving it the illusion of being performant without being able to generalize in practice[1]. The model might also have fatal weaknesses that can be exploited by an adversary to evade detection[2]. For the safety-critical environment of cybersecurity, the usage of such models is not ideal. In fact, black-box models are not even allowed in regulated fields unless they are supplemented with explanations [57], *e.g.,* courts do not consider model outputs as admissible evidence unless a forensic analyst is able to justify how the output links to the case [30]. Moreover, the "right to explanation" in the GDPR AI act also makes it tricky to deploy black-box models [58].

Explainable artificial intelligence (XAI)[3] has been proposed to open the proverbial 'black box' by making the model internals more human understandable [95]. The first mention of XAI can be traced back to van Lent *et al.* [124] in 2004, while the field really started growing drastically after DARPA announced its XAI program in 2014 [47].

In this paper, we systematize the increasingly growing microcosm of studies that develop and utilize XAI methods for security-specific target domains. We argue that the applications of XAI within cybersecurity are intrinsically different from other domains because cybersecurity works with practical use cases for safety-critical and high-stakes decision-making under adversarial settings. The lack of explainability is also an obstacle for deployment in cybersecurity [121]. In fact, explainability has arguably always been a core tenet in the design of ML pipelines for security [17], [111]. Even the seminal work by van Lent *et al.* uses XAI to explain the behaviour of AI-controlled entities in *military simulation games* [124].

In recent years, the security community has actively adopted XAI as a means to increase practitioners' trust [117]. Numerous studies have applied existing XAI methods to security applications [7], [12]. However, recent studies have recognized their shortcomings in addressing

1. This is often referred to as the Clever Hans phenomenon [116].
2. Adversarial machine learning studies these cases, see *e.g.,* [22], [106], [113].
3. The terms 'explainable artificial intelligence', 'explainable ML', and 'interpretable ML' are used interchangeably in the literature.

DOI 10.1109/EuroSP57164.2023.00022

the unique pain points of the security domain [107]. To this end, several security-specific XAI methods [60], [63], [138], and evaluation criteria [52], [129] have been proposed.

These recent developments have made XAI research within cybersecurity a fast-growing field: while there were only 42 articles about *'explainability'*, *'learning'*, and *'cybersecurity'* in 2015, that number has since skyrocketed to 2600+ in 2021, according to Google Scholar. This literature is fragmented across several research communities (including ML, security, graphics, and software engineering) with no unified overview. Additionally, existing works often use different terminologies interchangeably, *e.g.,* explicable, accountable, transparent, and understandable, making it more difficult to find relevant literature.

To the best of our knowledge, this is the first SoK on explainable ML for cybersecurity[4]. By taking a step back, we synthesize insights from the vast body of fragmented literature and identify open areas to stimulate further research in this field.

Following the XAI definitions set forth by Roscher *et al.* [112], we identify three cybersecurity stakeholders, *i.e., model users*, *model designers*, and *adversaries* who utilize XAI for four distinct objectives within the security literature: (i) XAI-enabled user assistance, (ii) XAI-enabled model verification, (iii) explanation verification & robustness, and (iv) offensive use of explanations. The interplay between the stakeholders, objectives, and the stages of a typical ML pipeline are given in Figure 1. Particularly, the stakeholders remain central throughout our discussions. We further categorize the literature *w.r.t* the targeted security domain (*e.g.,* intrusion detection), ML model, and XAI method. This taxonomy serves as a guide for finding related literature on XAI for cybersecurity.

After carefully reviewing 300+ papers, we found that XAI has been most commonly used for providing decision support to model users – 58% of the works are classified under *XAI-enabled user assistance*. User evaluation is a critical aspect of these studies to ensure that they are usable, and are sufficiently aligned with existing analyst workflows. However, user studies are conducted in only 14% of the cases, which is alarming since these methods aim to work directly with model users. We propose ideas for mitigating the lack of user studies in §9.

In addition, the stakeholders we identify have different competencies, and thus require tailored explanations [24], *e.g.,* model designers are typically experts in ML while model users are not. However, we identify several cases that either do not distinguish between model users and designers or do not specify any stakeholder. In contrast, model users and adversaries interact with the explanations in similar ways, but with opposing intent, requiring special manoeuvres to limit adversary access.

Furthermore, the role of model designers is substantial in cybersecurity for ensuring the security of the model and its explanations. Yet, the reviewed literature only provides decision support to model designers in 22.3% of the cases. We tease out the role of model designers in §8: we present a walk-through tutorial of how model designers can utilize XAI to detect and remove spurious features in a network attack detection scenario. The tutorial serves as a practical and easy starting point for security practitioners by showing three concrete ways to debug a black-box model via XAI. We show cases where the explanations are helpful, and cases where model designers may draw misleading conclusions instead. We discuss what can go wrong when explaining black-box models, and advocate for interpretability by design.

Organization: In §2 and §3, we describe the scope and the proposed taxonomy. In §4-§7, we elaborate on the main takeaways from the reviewed literature. In §8, we demonstrate how model designers can use XAI to debug their models. In §9, we present open problems and recommendations for further XAI research within cybersecurity.

2. Background and Methodology

Explainable machine learning. ML pipelines either use white-box models, which are inherently *interpretable*, or use black-box models that are explained via *post-hoc explainability*. For instance, linear regression and decision trees are considered white-box, while neural networks and random forests are considered black-box. The output of a post-hoc explainability method is either an *interpretable surrogate model* that approximates the black-box model, or is an explanation of the black-box model in terms of *model components* (*e.g.,* feature importance) or *input examples* (*e.g.,* counterfactuals). Additionally, an explanation can either elucidate how the model prediction is affected by a single data point (*local methods*), or by all data points (*global methods*). Note that interpretable models provide both local and global explanations. Furthermore, most post-hoc methods can be applied to any ML model, making them *model-agnostic*, while interpretable models are also referred to as *model-based explanations*. Figure 2 shows the various XAI terminologies, adapted from [2].

Method and Scoping. We synthesize available literature that uses XAI for (offensive and defensive) cybersecurity tasks. To this aim, we collect relevant literature, apply a reflexive thematic analysis [25] to construct a taxonomy based on common themes (*i.e.,* application objectives), and classify the literature into those themes. The literature was collected by seven researchers. Each paper was investigated by at least two researchers independently and discussed with all authors during weekly meetings. The code books were updated as new categories emerged.

We collected peer-reviewed literature that has used explainable models to address cybersecurity problems since 2014, *i.e.,* post-DARPA, by searching popular scientific repositories (*e.g.,* IEEE Xplore) and top security conference proceedings (*e.g.,* Usenix). We used known search terms, *e.g.,* '*explainable*', '*interpretable*', '*artificial intelligence*', '*cybersecurity*', '*robust*', '*offensive*', '*attacks*', and '*trustworthiness*'. To handle the fragmented literature, we expanded our search to include synonyms of the search terms at smaller security and non-security venues (see appendix A for the full list of venues). We also included older popular works that try to explain their model without explicitly using XAI, see *e.g.,* [17], [35], [40]. After carefully reviewing 300+ papers, we select 75 cybersecurity studies to build the taxonomy. Since it is impossible to

4. Although Hariharan *et al.* [65] present a short survey on XAI for cybersecurity, it covers only a small fraction of the literature. Moreover, the Explainable Security (XSec) framework proposed by Vigano *et al.* [125] is a non-conventional take on explainability, and does not embed the traditional XAI concepts within the security context.

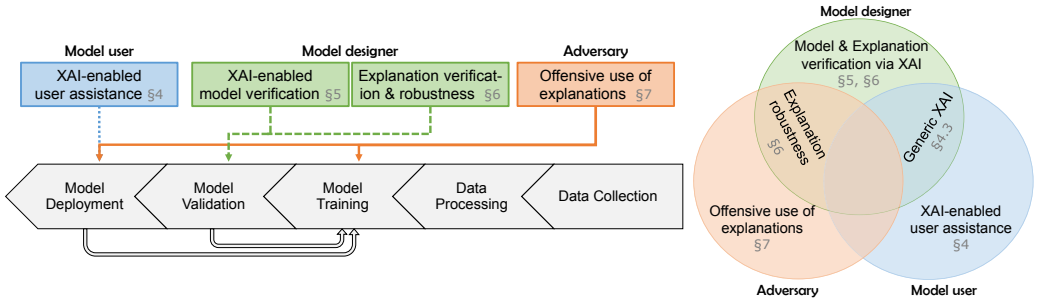

Figure 1. The interplay between application objectives and an ML pipeline (Left); and stakeholders and application objectives (Right).

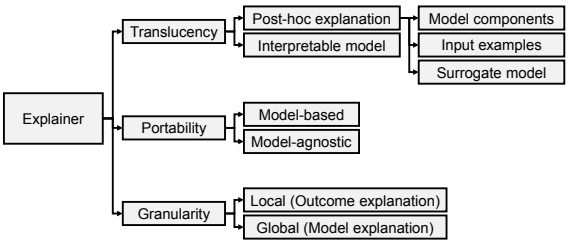

Figure 2. XAI ontology showing the key concepts within XAI.

cover all the available literature in the limited space, we chose representative works from each problem area. As such, there is some overlap with usable security, safety, and robustness literature, but we mainly focus on the use of XAI within cybersecurity.

3. Systematization

Given an ML pipeline from data collection to model deployment, explainable ML is applied once a model becomes available. In the literature, XAI has been used to explain the model output to a human, either for supporting them in decision-making or to understand whether the model works as intended. In addition, the adversarial threat landscape of cybersecurity suggests that XAI can also be used by an adversary to gain actionable information about the model in order to strengthen their attacks. This implies the existence of three stakeholders who interact with different phases of an ML pipeline and accomplish distinct objectives using XAI. We classify the literature based on the stakeholders, application objectives, target domain, model and explainer class. Figure 1 shows an overview of the stakeholder objectives that can be accomplished by applying XAI on a typical ML pipeline.

Stakeholders. We identify three stakeholders (explainees) who have different intents and expertise, and thus consume explanations for distinct objectives, even when interacting with the same ML model. The explanations are hence tailored to the specific needs of the stakeholders.

a) Model user is defined as a broad class of personnel who utilize the ML pipeline to improve the defence capacity of an organization, such as an analyst, developer, operator, domain expert, practitioner, or end-user. To this aim, a model user utilizes XAI techniques to better understand the output of a deployed model and make informed decisions, *e.g.,* a malware analyst uses explanations to gain insights into why a binary was classified as malicious [94].

b) Model designer is responsible for engineering the ML pipeline used for a security application, and consequently has a more intimate relationship with the model. A model designer utilizes XAI techniques during model training and validation to verify that the model works as intended, *e.g.,* a malware analyst uses explanations to investigate the causes of misclassifications, and to ensure that the model employs meaningful features [18]. Moreover, since the ML pipeline exists in an adversarial threat landscape, the model designer also ensures the safety and robustness of the model and its explanations [9].

c) Adversary is a human or an automated agent (malware) that intends to harm an organization by compromising the ML pipeline. An adversary exploits XAI techniques to formulate more efficient attacks, *e.g.,* by discovering weaknesses in the model [79]. In addition, an adversary may attack the XAI component of the ML pipeline itself to alter the generated explanations [56]. Depending on the attacker model, the adversary may interact with the explanations either during model training or after the model is deployed.

Application objectives. We classify the literature under four application objectives based on the intended use of the XAI technique — XAI is used to provide decision support to model users in (1); model designers in (2) & (3); and adversaries in (4).

(1) XAI-enabled user assistance covers techniques that are developed and utilized to support *model users* in making informed decisions, usually with the help of visual analytics dashboards. The explanations are meant to give control back to the user by helping them understand the model [96], and providing additional insights regarding the input data [28]. Since it is the model designers who typically develop the explanations for model users, it is essential to include model users during the evaluation process to understand the explanation efficacy.

(2) XAI-enabled model verification studies techniques that are developed and utilized to help *model designers* debug and validate the correctness of the ML model. These explanations are usually more technical in nature. In the literature, XAI has primarily been used to discover spurious/faulty features by investigating a given black-box model using, *e.g.,* feature importance [18], [32] or surrogate models [43], [63].

(3) Explanation verification & robustness studies techniques that are developed and utilized to help *model*

designers debug and validate the correctness & robustness of the XAI component in the ML pipeline. These methods focus on testing the correctness of post-hoc explanations under natural settings [86], and the robustness of explanations produced by post-hoc methods [9] and interpretable models [126] under adversarial settings [44].

(4) Offensive use of explanations studies how *adversaries* can exploit insights provided by XAI techniques for enhancing their capabilities, *e.g.,* i) by using explanations to compromise the privacy of the model, and ii) by using explanations to compromise the integrity and availability of the model. These attacks can be deployed in the model training phase (*e.g.,* poisoning attacks [118]), and model deployment phase (*e.g.,* privacy attacks [140]).

Target domain. We further classify the literature according to the cybersecurity target domain. In terms of defensive security domains, we cover: *malware detection*, *anomaly detection*, *intrusion detection*, *alert management*, *vulnerability discovery*, *asset prioritization*, *phishing detection*, *reverse engineering*, *traffic classification*, and *privacy protection*. In terms of offensive security domains, we cover: *privacy attacks* (*e.g., membership inference*, *model inversion*, and *model extraction*), *poisoning attacks* (*e.g., backdoor injection*), and *evasion attacks* (*e.g., test-time adversarial perturbations*). The papers that address generic non-security concepts, such as *anomaly detection* are further categorized according to the data sources they use, *e.g.,* image, binary, and network traffic. To the best of our knowledge, this is the first SoK to cover such a broad range of target domains.

Model & explainer class. Finally, we specify the ML models and XAI techniques used in the literature. The models are grouped according to the algorithm and the input data type accepted by the model (*e.g.,* tabular, images). The models are further classified coarsely as either black-box or white-box models, following the consensus of the ML community, see Table 1 for the model code book. The XAI techniques (called explainers henceforth) are categorized according to their underlying mechanism (*e.g.,* model components, examples, surrogate), see Table 2 for the explainer code book.

Table 3 provides a summary of the reviewed literature, which also showcases the co-occurrence of certain models and explainers. Note that the classification in Table 1 reflects the general level of understanding provided by the model class, while Table 3 shows the actual usage of the model: some studies explicitly treat white-box models as black-box for a model-free approach, see *e.g.,* [7], [41]. Other studies utilize an incomprehensible feature set (*e.g.,* by replacing feature names with integers), turning an interpretable model into a black box, see *e.g.,* [72].

The overview also helps us identify the misleading usage of certain terminologies. For instance, some works report their methods as being 'interpretable' while utilizing post-hoc explainers for black-box models, see *e.g.,* [63], [108], [139]. Strictly speaking, black-box models cannot be interpretable [112]. Thus, we categorize such works under post-hoc explainability. Note that it is possible to have an interpretable model that also uses a post-hoc explainer, but when a black-box model is explained via an interpretable model, it is called a surrogate model.

TABLE 1. CODE BOOK FOR ML MODEL CLASSES. 'W' REPRESENTS WHITE-BOX, AND 'B' REPRESENTS BLACK-BOX MODELS.

Model class	Machine learning algorithms	w	b
CNN	Convolutional neural networks for image data, *e.g.,* ResNet, VGGnet, RPN, and inception network		•
DNN	Deep neural networks for tabular data, *e.g.,* MLP, and auto-encoder		•
GNN	Graph neural networks, *e.g.,* GCN, and graph attention network		•
SeqNN	Sequential neural networks, *e.g.,* RNN, LSTM, and transformers		•
Kernel-SVM	Support vector machine with non-linear kernel		•
Ensemble	Ensemble of models, *e.g.,* random forest, gradient boosting trees, and neural network ensembles		•
LM	Linear models, *e.g.,* logistic (rule) regression, linear rank regression, and linear SVM	•	
RBC	Rule-based classifiers, *e.g.,* decision trees, regular expressions, and BRCG	•	
NB	Naive Bayes and its gaussian variant	•	
Automata	Abstract computing machines, *e.g.,* Markov chains, and probabilistic deterministic finite automata	•	
kNN	K-nearest neighbors	•	•
Unsupervised	Clustering algorithms, *e.g.,* HDBSCAN, kmeans, with(out) dimensionality reduction, *e.g.,* self-organizing maps, PCA, t-SNE	•	•

TABLE 2. CODE BOOK FOR EXPLAINER CLASSES.

Explainer class	Explanation methods
SHAP	SHAP and its variants, *e.g.,* kernelSHAP
LIME	LIME and its variants, *e.g.,* graphLIME
LEMNA	Non-linear LIME variant for security applications
GNNExplainer	Explanation method for graph neural networks
Grad-based	Gradient-based methods, *e.g.,* GradCAM, saliency map, integrated gradients, and layer-wise relevance propagation
Activation	Neuron activations, activation maps and attention
Importance	Feature importance computed using tree-based splitting, feature permutation, and SOM-based dimensionality reduction
Exemplar	Example-based explanations, *e.g.,* kNN, prototypes, protoDash
Contrastive	Contrastive explanations, *e.g.,* counterfactuals
Anomaly-score	Custom metric capturing deviation from normalcy, *e.g.,* decoder reconstruction loss
Visual-explanation	Explanation based on visualizing model components or model output for human perception
Sur-RBC	Surrogate rule-based classifiers, *e.g.,* decision trees, decision lists, and rule sets
Sur-Mixture	Surrogate mixture linear regression model
Sur-Automata	Surrogate automaton model

4. XAI-enabled User Assistance

The fundamental objective for employing (explainable) ML methods in security workflows is to provide decision support to model users. In fact, practitioners have been trying to make their models understandable since before the popularity of XAI, see *e.g.,* [17], [128], [137]. Over the past decade, numerous XAI applications have arisen to support model users in their decision-making when interacting with a deployed model. The prominence of this objective is evident from the distribution of the available literature — 58% of the reviewed studies provide decision support to model users.

Within the reviewed literature, the explanations are generated for distinct purposes at different levels of expertise even when considering a single stakeholder. For instance, to assist *software developers* in understanding vulnerable code, some approaches simply highlight the lines of code that the model thinks are vulnerable [46], [84], [115], while others extract human-understandable rules from the vulnerable code that can serve as actionable intelligence for periodic scanning and control [137], [141]. As such, these methods fall under two broad application scenarios: i) XAI is employed to provide assistance to model users for understanding model decisions and reducing their workload (*i.e.,* threat prioritization, false alarm

TABLE 3. SUMMARY OF XAI LITERATURE WITHIN CYBERSECURITY. ROWS ARE ORDERED *w.r.t* OBJECTIVES, TARGET DOMAIN, AND YEAR.

			Stakeholder			Objectives				Type		Models														Explainers												
Ref.	Year	Target domain	User	Designer	Adversary	User assist.	Model verif.	Explanation verif.	Offensive use	Interpretable	Post-hoc	CNN	DNN	GNN	SeqNN	Kernel-SVM	Ensemble	LM	RBC	NB	Automata	kNN	Unsupervised	SHAP	LIME	LEMNA	GNNExplainer	Grad-based	Activation	Importance	Exemplar	Contrastive	Anomaly-score	Visual-explanation	Sur-RBC	Sur-Mixture	Sur-Automata
[120]	2018	Alert management	•			•				✓							✓												✓					✓			
[98]	2021	Alert management	•			•			✓												✓																
[123]	2022	Alert management	•			•				✓				✓														✓									
[103]	2022	Alert management	•			•				✓							✓						✓														
[85]	2018	Anomaly detection (sensor)	•			•			✓												✓																
[26]	2018	Anomaly detection (syslogs)	•			•				✓				✓														✓									
[72]	2020	Anomaly detection (syslogs)	•			•				✓			✓	✓			✓		✓					✓	✓					✓							
[15]	2021	Anomaly detection (network)	•			•				✓			✓																		✓					
[14]	2021	Anomaly detection (sensor)	•			•			✓	✓					✓	✓	✓		✓			✓							✓								
[69]	2021	Anomaly detection (sensor)	•			•				✓							✓							✓													
[63]	2021	Anomaly detection (network)	•	•		•	•			✓		✓		✓	✓																						✓
[28]	2022	Anomaly detection (network)	•			•			✓												✓																
[83]	2021	Asset prioritization	•			•				✓		✓			✓										✓												
[57]	2021	Asset prioritization	•			•			✓									✓																			
[67]	2021	Asset prioritization	•			•			✓	✓									✓				✓										✓				
[122]	2020	Intrusion detection	•			•				✓			✓																					✓			
[127]	2020	Intrusion detection	•	•		•	•			✓			✓											✓													
[12]	2021	Intrusion detection	•			•				✓			✓											✓													
[91]	2021	Intrusion detection	•			•			✓										✓																		
[89]	2021	Intrusion detection	•			•			✓	✓								✓	✓											✓	✓						
[108]	2022	Intrusion detection	•			•				✓					✓														✓								
[96]	2021	Malware analysis	•			•				✓													✓										✓				
[17]	2014	Malware detection	•			•			✓									✓																			
[128]	2016	Malware detection	•			•			✓										✓																		
[10]	2017	Malware detection	•	•		•	•			✓							✓												✓				✓				
[60]	2018	Malware detection	•	•		•	•			✓			✓		✓																				✓		
[94]	2019	Malware detection	•	•		•	•			✓							✓								✓												
[92]	2020	Malware detection	•			•				✓			✓																						✓		
[80]	2020	Malware detection	•	•		•	•			✓			✓												✓												
[70]	2021	Malware detection	•			•				✓		✓																✓									
[133]	2021	Malware detection	•			•				✓			✓																✓								
[19]	2021	Malware detection	•			•			✓	✓			✓			✓	✓		✓			✓			✓					✓							
[74]	2021	Malware detection	•			•				✓		✓																	✓								
[76]	2022	Malware detection	•			•				✓			✓				✓	✓												✓							
[39]	2017	Phishing detection	•			•			✓									✓																			
[31]	2021	Phishing detection	•			•				✓		✓			✓														✓								
[87]	2021	Phishing detection	•			•				✓		✓																					✓				
[59]	2021	Privacy protection	•			•				✓		✓					✓								✓			✓									
[35]	2010	Protocol analysis	•			•			✓												✓																
[49]	2020	Protocol analysis	•			•			✓												✓																
[102]	2017	Test-time perturbations	•			•				✓		✓																✓					✓				
[137]	2015	Vulnerability discovery	•			•			✓										✓																		
[115]	2018	Vulnerability discovery	•			•				✓							✓												✓								
[46]	2019	Vulnerability discovery	•			•				✓			✓																✓								
[141]	2020	Vulnerability discovery	•			•				✓		✓																		✓							
[7]	2020	Vulnerability discovery	•			•				✓								✓					✓		✓												
[84]	2021	Vulnerability discovery	•			•				✓				✓												✓											
[132]	2021	Vulnerability discovery	•			•				✓							✓								✓						✓						
[41]	2021	Uncertainty estimation	•			•				✓												✓								✓							
[36]	2017	Binary analysis		•			•			✓					✓													✓				✓				
[93]	2018	Intrusion detection		•			•		✓	✓			✓					✓													✓					
[18]	2020	Malware detection		•			•			✓		✓			✓																			✓		
[138]	2021	Malware detection		•			•			✓			✓																		✓					
[43]	2022	Malware detection		•			•		✓	✓			✓			✓	✓			✓		✓													✓		
[34]	2021	Privacy protection		•			•			✓							✓							✓												
[4]	2020	Traffic classification		•			•			✓		✓																	✓							
[32]	2021	Vulnerability discovery		•			•			✓			✓	✓												✓		✓								
[131]	2021	Anomaly detection (sensor)		•				•		✓													✓						✓							
[86]	2020	Backdoor injection		•				•		✓		✓													✓			✓								
[1]	2020	Intrusion detection		•				•	✓	✓							✓			✓				✓												
[3]	2018	Reverse engineering		•				•		✓		✓	✓															✓								
[9]	2018	Test-time perturbations		•				•	✓	✓		✓					✓	✓						✓	✓			✓								
[56]	2019	Test-time perturbations			•			•		✓		✓																✓		✓						
[44]	2019	Test-time perturbations			•			•		✓		✓																✓								
[139]	2020	Test-time perturbations			•			•		✓		✓																✓								
[78]	2020	Test-time perturbations			•			•		✓			✓															✓								
[51]	2021	Test-time perturbations			•			•		✓		✓																✓								
[55]	2021	Test-time perturbations			•			•		✓		✓													✓			✓								
[136]	2021	Backdoor injection			•				•	✓				✓											✓		✓									
[118]	2021	Backdoor injection			•				•	✓			✓				✓	✓						✓												
[79]	2021	Membership inf. & model steal.			•				•	✓			✓																		✓					
[119]	2021	Membership inference			•				•	✓		✓	✓												✓			✓								
[140]	2021	Model inversion			•				•	✓		✓																✓								
[42]	2019	Test-time perturbations			•				•	✓		✓																✓								
[134]	2021	Test-time perturbations			•				•	✓			✓															✓								

reduction, user awareness), or ii) XAI is employed for the synthesis of new information (*i.e.,* expert knowledge creation, reverse engineering). Below we provide examples from the literature showing the different uses of explanations for different model users.

Triaging and threat prioritization. *Security practitioners* receive an enormous influx of cyber data that needs to be analyzed. XAI-enabled triaging techniques have been proposed to reduce analyst workload by redirecting their attention to critical events. This is crucial for Security Operations Center (SOC) analysts who often suffer from 'alert fatigue' caused by investigating large volumes of intrusion alerts on a daily basis [98]. Black-box methods can often not be applied since the analysts are under contractual obligation to review all alerts [99]. Instead, XAI techniques can reduce their workload via intelligent alert management while enabling them to justify the model's decisions. The intuition is that a security analyst can use the explainable ML model as a 'virtual assistant' that discovers meaningful patterns in large datasets and presents them to the analyst who can then make informed decisions about which data to triage and what actions to take next [67]. For example, Nadeem *et al.* [98] propose alert-driven attack graphs that show attacker strategies learned from intrusion alerts. They utilize an interpretable suffix-based automaton model to learn contextually meaningful attack paths. *Security analysts* can triage critical alerts by selecting one of the attack graphs. van Ede *et al.* [123]

use an LSTM to learn the contextual meaning of alerts by capturing the correlation between them in an attention vector. Their system clusters attention vectors, capturing attack campaigns. *Security analysts* only need to analyze outlier and sampled events from emerging clusters, drastically reducing their workload. Similar approaches have been proposed to triage critical syslog entries for the forensic analysis of cyber attacks in a federated learning setup [108], and to efficiently allocate cyber resources for advanced persistent threat (APT) detection [83].

False alarm reduction. XAI can help *security practitioners* and other *model users* quickly disregard false alarms by explaining why the model made a prediction. For instance, Sopan *et al.* [120] propose a visual analytics dashboard to understand why an alert was raised. The dashboard provides an explanation of the alert in the form of an approximated decision path followed by the model and a list of important features. A similar approach is proposed in [91], [122]. Other works only show feature importance to help *security analysts* understand model predictions, *e.g.,* for malware detection [19], [74], [76], and anomaly detection [12], [14], [26], [69], [72]. In contrast, instead of explaining the predictions, de Bie *et al.* [41] have proposed a metric to help *security analysts* weed out false or untrustworthy predictions in regression models. They follow the intuition that instances close to each other typically have similar predictions. Thus, by comparing the prediction of a given instance with those of its k-nearest neighbours, a *model user* can identify whether the prediction can be trusted.

A handful of works have used anomaly scores to automatically discard anomalous events, thus reducing the cognitive load on *general model users*. For instance, Ardito *et al.* [15] use anomaly scores to support *medical staff* in detecting when an attack has occurred on a patient's e-health telemonitoring device. The auto-encoder-based system avoids processing anomalies that can otherwise have devastating effects on a patient's health. Instead, it sends out a validation request to the medical staff. Similarly, Akerman *et al.* [6] use image reconstruction loss as an indication of whether artefacts in ADS-B video frames are false alarms. ADS-B is a protocol used by air traffic controllers to communicate with pilots regarding surrounding objects. By highlighting what might be false alarms, *pilots* can efficiently focus on the mission at hand.

User awareness & education. XAI has been utilized to increase the general awareness of different *model users* for insecure behaviour deterrence. For example, to keep *Android users* safe, multiple works display warning signs with explanations for why an app was blocked or marked as malicious. The explanations are constructed from influential features extracted from apps' permission usage [17], [133], and network traffic [128].

XAI has also been used for warning *end-users* when they land on potential phishing websites to improve their overall Internet browsing behaviour [31], [39], [87]. For instance, Phishpedia [87] employs logo detection to generate visual explanations in the form of insightful annotations on the websites. Chai *et al.* [31] take one step further by developing a multi-modal learning setup for more accurate phishing website detection. Their attention-based explanations highlight the URL characters, website text, and images that were relevant for the detection.

Finally, in order to raise awareness among *security analysts* regarding the impact of adversarial examples on a given ML model, Norton *et al.* [102] develop a visualization suite that lets them investigate the effect of various gradient-based adversarial attacks on image classifiers.

Expert knowledge creation. XAI can be used to synthesize human-understandable knowledge from black-box models. For instance, Mahdavifar *et al.* [92] extract a surrogate rule set from a pre-trained neural network that substitutes the knowledge base of their expert system. *Security analysts* interact with the expert system, which uses the rule sets to explain classification decisions. These rules are then used to classify unseen security incidents (*e.g.,* malware attacks and phishing attempts).

In order to address the lack of interpretability in vulnerability discovery methods [81], Zou *et al.* [141] and Yamaguchi *et al.* [137] extract human-understandable rules from code snippets that the model thinks are vulnerable. These rules are then used by *software developers* to detect vulnerabilities in previously unseen code bases. Next to this, counterfactual explanations have been used to automatically generate patches for vulnerable code. Wijekoon *et al.* [132] discover vulnerabilities in source code and proactively correct them with the minimal changes necessary. To this end, they use LIME to find the nearest unlike neighbour as the most similar code snippet that is not vulnerable, which is then used as a patch.

Reverse engineering. Reverse engineering is commonly used in software engineering to convert black-box systems into white-box alternatives. However, there is a key difference between surrogate model learning and reverse engineering: the former extracts an interpretable model from a *black-box model*, while the latter either learns an interpretable model or uses a post-hoc explainer to *provide insights into the input data*. In this sense, reverse engineering methods can be considered as standalone tools that provide decision support to *model users* regarding input data.

The most common application of reverse engineering is to consider a live system as a black box, collect traces from it, and learn an interpretable model from these traces. This model can be relatively easily visualized for model-based explanations about the black-box system. For instance, Fiterau *et al.* [49] apply protocol state fuzzing on servers that use the Datagram Transport Layer Security (DTLS) protocol in order to discover functional and non-conformance issues in several implementations. Discoverer [40] and Prospex [38] are two other popular systems for reverse engineering application-level specifications of network protocols. Similarly, Cho *et al.* [35] learn an automaton from botnet traffic to understand its Command and Control (C&C) channels; Lin *et al.* [85] learn an automaton from sensors of a water treatment plant to detect potential sensor malfunction, and Cao *et al.* [28] learn an automaton from the network traffic of a Kubernetes cluster to identify misbehaving pods.

Alternatively, a black-box model can be learned from the traces, and post-hoc explainers can be used to explain the properties of the traces. For instance, Gulmezoglu *et al.* [59] want to understand the type of web requests that leak side-channel information, such as performance counters and cache occupancy. This leakage enables website fingerprinting attacks in which users can be tracked

by monitoring the unique combination of websites visited by their browsers. To this aim, they collect side-channel information leaked from different browsers, use it to learn several ML models, and use LIME and saliency maps to identify the leakiest web requests. Similarly, *Malware analysts* can use reverse engineering to understand the relationship between malware samples. For instance, Nadeem *et al.* [96] build behavioural profiles of malware samples by clustering their network activities. They visualize the overlap in the malware profiles in order to discover interesting malware capabilities. Similarly, Iadarola *et al.* [70] use gradient-based saliency maps to construct cumulative heatmaps for individual malware families that show visual differences between their disassembled code.

4.1. The Role of Visualizations in XAI

Visual explanations are the most common way to explain the inner workings of a black-box model. This is because human cognition prefers visual information over text for providing decision support [104]. The reviewed literature proposes several types of visual explanations, *e.g.,* a graph-based interpretable automaton model proposed by Lin *et al.* [85] that can directly be visualized for anomaly detection, and a context-based visual analytics dashboard proposed by Alperin *et al.* [7] that uses LIME and t-SNE for triaging vulnerabilities.

Visualizing the structure of tree-based models is another popular explanation method [10], [120], [122]. Sopan *et al.* [120] report that the security analysts found their visualization of an approximated decision path generally helpful. However, this is not always the case. Angelini *et al.* [10] propose a visual analytics system to explain the reason for malware detection by showing geo-locations of downloaded files and allowing a *malware analyst* to drill deeper into the individual paths of a random forest. However, simultaneously exploring the paths of $\sim$100 decision trees does not make it any easier to decipher what the model is doing. Instead, it is preferable to provide different explanations based on the user's trust, *e.g.,* by providing less explanation when trust is high, and more explanation when trust is low [11].

Usability is an important consideration when designing decision-support tools for human analysts. Every explanation method has an associated cost in terms of its adaptation time. Even a simple XAI tool that plots reconstruction errors and lists top-k anomalies can cost analysts a full day to get used to [12]. *Generally, simpler explanations are preferred, otherwise they can make the original task even more time-consuming* [105]. In other terms, complex visualizations contribute to cognitive load, subverting effective explanations. This is why the knowledge of existing analyst workflows is an important predictor in the successful deployment of XAI tools [103].

In addition, visualizations are not always equivalent to effective explanations. A model does not become interpretable just by virtue of visualizing it. For instance, the automaton model presented in [85] requires some level of expert knowledge to correctly interpret it. Similarly, the decision tree proposed in [91] is claimed to be interpretable by default since it mimics human-level decision making, while it does not appear to be size-limited to actually be considered interpretable [88]. Furthermore, the explanations provided by DeltaPhish [39] are incomplete because they are limited to the linear coefficients of a single SVM, while it uses an ensemble of SVMs for different features.

> **Takeaway 1:** *Visualization is not equivalent to effective explanation. XAI should reduce complexity, not add another layer of complex visualizations.*

4.2. Explanation Evaluation via User Studies

XAI-enabled user assistance tools can be evaluated along several dimensions, *e.g.,* fidelity, understandability, efficiency, and construction cost [64]. Most of these criteria can be evaluated without human involvement. However, understandability involves multiple usability factors that can only be suitably evaluated with model users. This is tricky because analyst time is expensive [63]. Thus, many existing works focus on evaluating other aspects of explanations instead, *e.g.,* their fidelity and efficiency [12], [127], [131]. However, an explanation is unlikely to be used in practice if it is not understandable, even if it is robust and correct. Therefore, we advise bringing humans back in the loop by evaluating XAI-enabled user assistance tools with application-grounded (with experts) or human-grounded (with lay persons) user studies [45].

In order to subvert costs, qualitative analyses are often conducted in place of user studies [10], [84], [98], [137], [138]. This is problematic because of the involvement of multiple stakeholders — the XAI tools are typically developed by *model designers* for *model users*. In practice, these stakeholders have different expertise. We recommend avoiding qualitative analyses that only investigate the happy flows (successful explanations) in order to circumvent the possibility of cherry-picking [82].

> **Takeaway 2:** *User studies are necessary to evaluate the usability of decision support tools. Yet, only 14% of the reviewed literature performs user studies with a median of 8 participants.*

4.3. The Importance of Stakeholder Specification

We identified six cases within the reviewed literature where the roles of model users and designers were entangled [10], [60], [63], [80], [94], [127]. These methods assume that the same person is both, the designer and the user: 1) Angelini *et al.* [10] propose a visual analytics system for helping *malware analysts* handle 'grey cases' where a model produces a classification with low confidence. The intuition is that the explanations can either enhance the analyst's confidence in the system if the explanations make sense, or *can trigger model improvement if they do not*. 2) Kyadige *et al.* [80] and 3) Mathews [94] explain the output of a malware detector to help *analysts* understand why a binary was classified as malicious, and *evaluate whether the model uses meaningful features.* 4) Wang *et al.* [127] use SHAP to help *security analysts* recognize the relationship between specific features and attack types, which *can guide the design of a more efficient intrusion detection system*. 5) LEMNA [60] and 6) DeepAID [63] are specialized XAI methods that address the unique challenges of the cybersecurity

domain, *e.g.,* non-linear decision boundaries and concept drift [97]. LEMNA is a non-linear variant of LIME, while DeepAID learns a surrogate automaton model that allows users to understand the black-box model and *improve it, if necessary*.

We also identified 17 cases where the intended stakeholder was left unspecified [1], [4], [14], [19], [26], [32], [34]–[36], [39], [49], [59], [74], [76], [83], [115], [131]. These methods appear to heavily focus on the fidelity of the explanations instead of their understandability, removing the human from the loop and potentially limiting their deployability.

Disentangling stakeholders is necessary for assessing how the proposed method translates to industry, since model users and designers are often distinct parties, working in different departments or even different organizations. Moreover, since model users and designers interact with distinct phases of the ML pipeline, they often have different expertise and require disparate explanations. For example, while both Russell *et al.* [115] and Chakraborty *et al.* [32] use activation to highlight vulnerable code, the former is intended for *model users* (explaining why a code snippet was considered vulnerable), and the latter is intended for *model designers* (making sure the model highlights correct code snippets). Even within the same domain and for the same stakeholder, the explanations can have contrasting uses, *e.g.,* within malware detection, some works use explanations to warn *smartphone users* of malicious apps on their phones [17], [128], [133], while other works provide more technical explanations to *malware analysts* regarding classifier decisions [19], [74], [76]. Thus, explanations meant for one type of user might be too vague or too technical for another user [24].

Takeaway 3: *Effective explanations are tailored to a specific user. Model users and designers usually have different expertise, and thus require disparate explanations. We encourage the community to specify their intended explanation stakeholders.*

5. XAI-enabled Model Verification

In fields other than cybersecurity, humans interact with AI systems with the assumption that they are near-perfect [100]. Thus, *faith* or *fidelity* is a major constituent of trust in the beginning, which is eventually replaced by reliance and predictability. The reverse is true for the adversarial threat landscape of cybersecurity: reliance and predictability are important constituents of trust since these systems can be attacked. To this aim, *model designers* have a vital role in validating the safety and correctness of the ML pipeline in order to build trust with practitioners.

A defensive security model designer is generally concerned with two aspects of model verification: (i) the model is robust to adversarial perturbations, and (ii) the model is generalizable and works as intended. The former is covered by the adversarial learning literature that aims to limit the possibility of evasion by making models robust to adversarial perturbations [22], [113]. Recent works have started to investigate the relationship between robustness and interpretability — early evidence suggests that robust models may be more interpretable than their non-robust counterparts [114]. The intuition here is that robust models are smoother and can thus be more easily interpreted by humans. Nevertheless, further research is warranted to explore how XAI can guide the search for tamper-proof features used to train robust models.

The latter aspect of model verification fundamentally scrutinizes the generalizability of the model. Generalization is a highly desirable property in learning-based security systems as they are meant to detect previously unseen threats. To this aim, XAI has been used to detect spurious correlations — artefacts unrelated to the security task that allow the learning algorithm to create shortcuts for separating the classes, instead of actually solving the task. These artefacts make the model *seem performant* without being able to generalize in practice [16], [42]. In this systematization, we expand the traditional definition of spurious features to also include faulty features whose distributions are not representative of real-world cases [97]. In the literature, spurious/faulty feature detection is done via conformance checking, influential feature analysis, and surrogate model analysis.

Conformance checking. Comparing classifier decisions with some notion of ground truth can be used for model debugging. For instance, Kyadige *et al.* [80] and Chua *et al.* [36] compare model outputs with expert knowledge as a sanity check to ensure that the model works correctly. Specifically, given an RNN that recovers function types and signatures from decompiled binaries, Chua *et al.* [36] use post-hoc explainers to verify that the model is able to learn concepts comparable to an expert's domain knowledge. To this aim, they use t-SNE to visualize semantically similar word embeddings, and saliency maps to understand which instructions are relevant for the recovery of the input functions. Nevertheless, many security applications struggle with obtaining ground truth, making conformance checking difficult in practice [97].

Influential feature analysis. Feature importance can be employed to investigate whether the model uses meaningful features. For example, Chakraborty *et al.* [32] use LEMNA to check whether the highlighted tokens meaningfully communicate why a code snippet was classified as vulnerable. Similarly, Reyes *et al.* [1] use SHAP and Ahn *et al.* [4] use feature permutation to select meaningful features for intrusion detection and network traffic classification, respectively.

Feature importance can also be used to investigate causes of misclassifications in order to improve the model. For example, Becker *et al.* [18] propose a visual analytics system that enables *malware analysts* to explore how the model views malware samples at different layers by clustering neuron activations. By visualizing the internal components of black-box models, malware analysts can identify sources of bias and misclassifications. Another example is from the domain of voice assistants: Chen *et al.* [34] design a more robust voice assistant by first using SHAP to identify the type of fuzzy words that cause a given tree ensemble-based wake-up word detector to become falsely triggered, and then proposing countermeasures to avoid it from happening. Within continual learning settings, CADE [138] explains the cause of performance degradation of a malware detector by reporting the features that are most affected by concept drift. It uses the contrastive explanation method: it perturbs features to see which combination increases the distance to the

training data the most and thus is responsible for causing drift. Finally, Marino *et al.* [93] identify and correct the cause of missed detections and false alarms in IDS. They use adversarial examples that naturally serve as counterfactual explanations, showing the minimal changes required in feature values to correctly classify the (misclassified) security events. They expect that these insights will further improve their IDS performance. However, it is unclear how they avoid over-fitting since they utilize the knowledge of the test set to improve their model performance.

Surrogate model analysis. Interpretable surrogate models can be inferred from black-box models that can directly be inspected for defects, *e.g.,* Han *et al.* [63] learn a surrogate automaton model, while Dolejš *et al.* [43] learn a rule-based surrogate model. In addition, Dolejš *et al.* [43] measure the interpretability of the surrogate model in terms of its behavioural similarity to the black-box model, *i.e.,* by checking whether they make similar mistakes.

5.1. The Risks of Post-hoc Explainability

As it stands, the security literature heavily relies on performance metrics (*e.g.,* F1 score) as a means to conduct model verification. Goodhart's law dictates that when a measure becomes a target, it ceases to be a good measure [37]. This is evident from the abundant literature on adversarial learning, suggesting that merely relying on performance metrics is a dangerous strategy as the model might have fatal weaknesses that an adversary can exploit. Moreover, high performance on experimental data does not imply that these methods would generalize in practice. This is because the analysis is rarely conducted in operational settings due to excessive costs. Furthermore, security papers often skip details on the operationalization of the ML pipeline, making it difficult to know if any spurious/faulty features have been used. As such, feature attribution can be used for identifying spurious features [16]. In fact, spurious feature detection and removal should become more commonplace before deploying new models. We also recommend that publicly available models be supplemented with a verification analysis to enhance trust among practitioners. To assist *model designers*, we provide an illustrative walk-through of how they can debug their models using commonplace XAI tools in §8.

Having said that, model designers must also be aware of the risks of using post-hoc XAI for model verification: post-hoc explanations are approximations of black-box models that either hide away details or learn different concepts altogether. For example, explanations based on feature importance often disagree on the same model prediction [77], suggesting that there is a mismatch between the explanations and what the model actually does. A similar observation has been made for surrogate models [5]. In fact, it is even possible to extract fair explanations from known unfair models. In regulated environments where companies are required to supplement their black-box models with explanations, they can be abused to perform *'fairwashing'* — promoting the false perception that a model is fair when it is actually not [5]. Therefore, it is advisable to opt for interpretable models. Where that is not possible, it is critical to establish an equivalence relationship between a model and its explanation, *e.g.,* by learning a certifiably equivalent surrogate model. For instance, Weiss *et al.* [130] and Koul *et al.* [75] extract equivalent deterministic finite automata from black-box RNNs. These works fall under the safety verification literature, see *e.g.,* [8], [54], [68], [135].

> **Takeaway 4:** *Regardless of the XAI method used to validate a model, it is vital for safety-critical applications to establish an equivalence relation between the model and its explainer. However, this is not yet common practice within cybersecurity.*

6. Explanation Verification & Robustness

Whilst using XAI for model verification, the explanations themselves need to be verified for correctness and robustness. *Model designers* are thus also responsible for conducting explanation verification to ensure the safety of the ML pipeline. This is an important line of work because XAI methods can sometimes trigger on input data patterns rather than on meaningful model behaviour. For instance, Adebayo *et al.* [3] reset the weights of a neural network to their initial random values and show that some gradient-based methods still use information from the input. Therefore, evaluating the fidelity of explanations becomes vital. Yet, it has sometimes been overlooked within the security literature, see *e.g.,* [18], [122]. In addition, qualitative analysis alone does not provide sufficient test coverage, and may even lead to cherry-picking [82]. Knowing that XAI methods can generate arbitrary explanations, objective evaluation criteria are required to ensure that (a) the explanation methods work [3], [86], [131], and (b) the explanations are robust to adversarial attacks [9], [27], [33], [50], [66], [126]. Warnecke *et al.* [129] and Ganz *et al.* [52] are excellent starting points for evaluation criteria for a wide variety of post-hoc explainers under security settings. Their criteria include descriptive accuracy, sparsity, completeness, stability, efficiency, and robustness.

Fidelity evaluation. The explanation fidelity can be evaluated in several ways: Wickramasinghe *et al.* [131] test the fidelity of their attribution method by perturbing feature values and analyzing their impact on the explanations. Lin *et al.* [86] test the correctness of different saliency explanations by deliberately injecting artefacts in the input data to see if the explanations detect them. Specifically, they inject backdoor trigger patterns in input images that would naturally result in misclassifications by a CNN. These backdoor triggers serve as ground truth, *i.e.,* the backdoor features are primarily responsible for causing misclassifications, so a faithful explainer must be able to identify them.

Adversarial robustness. More importantly, XAI forms an additional attack vector for *adversaries* within the context of cybersecurity — both post-hoc explainers [44], [51], [55], [56], [78], [139] and interpretable models [21], [90] are sensitive to small adversarial perturbations. For instance, Ghorbani *et al.* [56] investigate the effect of adversarial perturbations on exemplars. Exemplars are samples from the training set whose features most resemble the instance to be explained. They find that while keeping the prediction equal, they can cause the top-

3 exemplars to be entirely different for perturbed samples, implying that the perturbed samples enter a part of the model with drastically different latent features. Moreover, Dombrowski *et al.* [44] exploit the fragility of explanations to perform targeted attacks. They show that by adding imperceptible perturbations to the input image, the adversary can completely control the generated explanation. These studies identify three problematic traits of post-hoc explainers:

- Predictions and explanations can change tremendously under small perturbations;
- While keeping the explanation fixed, input samples can be perturbed to cause misclassifications;
- While keeping the prediction fixed, input samples can be perturbed to change the explanation.

The fact that models and explainers can be attacked independently opens up a new range of attacks. For instance, malware authors can evade detection while masking the features that they used for evasion. In this case, the generated feature importance maps do not represent the features that are actually important for classification. Therefore, it is imperative that *model designers* robustify explainers against adversarial perturbations.

Recent works have started to investigate the robustness of post-hoc explainers: Alvarez-Melis *et al.* [9] investigate the smoothness of explanations around data points as a measure of robustness. Based on a local version of the Lipschitz constant, they show that the smoothness of model-agnostic explainers, *e.g.,* LIME and SHAP, can vary across datasets. They also show that gradient-based explanations are approximately four times smoother than LIME, suggesting that model-based explanations are more robust than their model-agnostic counterparts. For counterfactual explanations, Fokkema *et al.* [50] show that robust explainers cannot also be recourse sensitive[5]. This means that there will always be model inputs for which the explanations suggest modifications that do not end up changing the model's prediction[6]. As a solution, they suggest using multiple counterfactual explanations pointing in different directions.

A handful of works have also proposed robust variants of interpretable models, such as linear models and decision trees. For instance, Vos *et al.* [126] learn efficient and robust decision trees, while Hayes *et al.* [66] learn robust and differentially private logistic regression. Note that decision trees and logistic regression are considered interpretable as long as they are size-limited [88].

Takeaway 5: *Along with ML models, explainers can also be attacked. In addition to fidelity testing, we recommend either using a robust explainer or conducting explanation verification under adversarial settings.*

7. Offensive Use of Explanations

From the offensive security perspective, XAI can also provide decision support to *adversaries* for better attack formulation. As outlined by Papernot *et al.* [106], adversaries can target multiple phases of an ML pipeline, *e.g.,* the training phase for poisoning attacks, and the deployment phase for evasion and privacy attacks. XAI can further strengthen these capabilities by exposing sensitive details about the model. Adjusting the definitions proposed in [106] for XAI, we organize the nefarious uses of explainers through the lens of the classical confidentiality, integrity, and availability (CIA) triad [109]. Considering the added utility of XAI, attacks on *confidentiality* utilize explanations to expose the model structure or the data on which the model was trained. Attacks on *integrity* and *availability* utilize explanations to discover knowledge that adversaries can use to induce specific model outcomes of the adversary's choosing and thwart legitimate users from accessing meaningful model outputs.

Confidentiality attacks. Explanations provide additional information to *adversaries* about the inner workings of a deployed model, making it easier to reconstruct the model and the training data. This is why explanations are seen as privacy vulnerabilities [62]. Yet, little work is done to generate privacy-preserving explanations [23]. In the literature, XAI has been used to strengthen model inversion [140], membership inference [119] and model extraction attacks [79].

Model inversion attacks enable adversaries to reconstruct training data from model predictions [106]. Adversaries can reproduce the model more accurately by utilizing explanations, *e.g.,* Zhao *et al.* [140] use an XAI-aware model inversion attack to successfully recover images from the training data. They show that feature importance maps generated from gradients and layer-wise relevance propagation (LRP) helped improve image reconstruction and led to an increase in model inversion performance compared to only using predicted probabilities.

Membership inference attacks assume that an adversary has some inputs and they want to predict whether they were used during training [106]. Shokri *et al.* [119] utilize gradient-based explanations to perform stronger membership inference attacks. They use the variance of saliency maps as a feature to infer membership and show that it works better than mere random guessing. The performance further improves when using the full explanation instead of only the variance.

Finally, model extraction attacks enable adversaries to recover the model's structure and parameters from predictions [106]. Kuppa *et al.* [79] utilize counterfactual explanations to improve their model extraction and membership inference attacks. They perform the model extraction attack by learning a surrogate model from known predictions and explanations. In addition, they perform membership inference by comparing the predictions of the target and counterfactual models to infer whether an input belonged to the training data.

Integrity and Availability attacks. Explanations provide additional knowledge to *adversaries* about the features to perturb in order to alter the correct functioning of the model. This can be done while the model is already deployed (*i.e.,* evasion attacks) or when the model is training (*i.e.,* poisoning and backdoor attacks).

Demetrio *et al.* [42] use integrated gradients to explain the importance assigned to the various fields of binary executables. They use this information to identify a few

5. Recourse refers to a description of feature modifications required to change the model outcome.

6. Note that the inputs for which this happens might not occur in practice and that this problem does not exist in linear unbounded models.

bytes in the malware header that need to be perturbed in order to successfully evade detection.

Poisoning attacks are specialized adversarial attacks where an adversary injects a small percentage of perturbed data to get some desired change in the learned model. Kuppa *et al.* [79] use counterfactual explanations to find the malware features that most heavily impact the classifier decision. They use this knowledge to craft adversarial training samples that efficiently poison the model.

Backdoor attacks are specialized poisoning attacks where the adversary makes the model sensitive to a pre-specified trigger. Severi *et al.* [118] use SHAP to craft backdoor triggers in malware detectors. Utilizing the explanation, they determine which features to poison, resulting in a success rate of up to three times higher than that of a greedy algorithm that does not use XAI. Similarly, Xu *et al.* [136] inject backdoors into GNNs by leveraging XAI techniques. They employ GNNExplainer to identify the parts of the graph to attack, and GraphLIME to identify the node features and values to change.

In two-player competitive games, Wu *et al.* [134] utilize XAI to exploit the weakness of an adversarial reinforcement learning agent. In such games, the agents take optimal actions according to their policy function, which is often learned using self-play. Using saliency maps, the proposed adversarial agent observes which of their actions the opponent pays the most attention to, and alters them in the next time stamp, thus confusing and manipulating the opponent's actions.

Takeaway 6: *Uniquely attributed to the security domain, adversaries may abuse explanations to bolster their capabilities. Meanwhile, research on privacy-preserving explanations that are also robust to evasion attacks is almost non-existent.*

8. Tutorial: Debugging a Malicious Network Traffic Detector via XAI

Sections §5 and §6 elucidate the critical role of model designers in ensuring the correctness and robustness of an ML pipeline. While XAI has been commonly directed towards model users, we argue that model designers can also greatly benefit from it. In this section, we present an illustrative tutorial on how model designers might use XAI for model verification. Specifically, we demonstrate how the investigation of influential features and misclassifications can identify problematic or spurious features. The experiments necessitate a sufficient understanding of post-hoc explainers for correct interpretation and highlight the expressive power of interpretable models.

We consider a *model designer* who learns an ML model to detect malicious botnet traffic on their company's network. They have some intuition of how a potential botnet-infected device might behave, and thus use XAI to validate whether the model follows that intuition, *e.g.,* by checking whether it uses any spurious features or any strange artefacts from the training data. Note that we are only interested in finding spurious features: while the selection of tamper-resistant features is also an important problem, using XAI to discover such features remains an open problem, to the best of our knowledge.

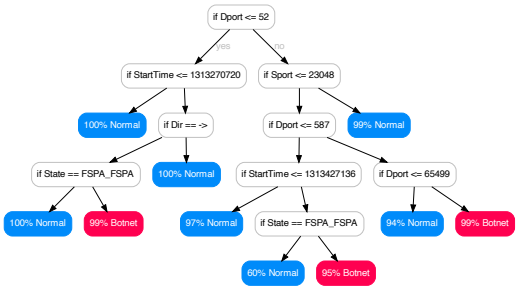

Figure 3. Decision tree for the CTU-13 dataset. It uses StartTime and Sport to differentiate between benign and malicious Netflows.

Dataset selection. We use the open-source CTU-13 [53] as our experimental dataset. It has 13 scenarios, each containing both benign and malicious Netflow data. The malicious Netflows in each scenario are collected by monitoring virtual machines (VM) infected with real malware. Each Netflow has the following features: start time (StartTime), duration (Dur), protocol (Proto), source port (Sport), Netflow direction (Dir), destination port (Dport), state (State), source type of service (sTos), destination type of service (dTos), total packets (TotPkts), total bytes (TotBytes), and source bytes (SrcBytes). The dataset contains 64,855,215 benign and 1,535,374 malicious Netflows.

Experimental setup. We have developed a modular XAI pipeline in Python with six models and four explainers. Implementation details are given in appendix B. We release the code for reproducibility[7].

For the experiments, we train a gradient boosting machine (GBM) over all the features of the Netflow data, as described in [53]. The GBM achieves a balanced accuracy of 86.4%. While this model is arguably not state-of-the-art for detection purposes, it is a black box that concretely shows how improvements can be obtained via XAI. As such, the analysis described in this section can be applied to any black-box model[8].

We use SHAP, LIME and LEMNA to explain the predictions of the GBM. We also learn an interpretable decision tree (see Figure 3) to verify whether similar conclusions can be drawn from model-based and model-agnostic explanations. The decision tree has nine nodes and achieves a balanced accuracy of 83.6%, which is only slightly worse than the GBM. The SHAP summary plot and LEMNA explanations[9] for the GBM are in appendix D. We generate explanations for 140 Netflows from the test set: 50 true positives (malicious), 50 true negatives (benign), 20 false positives (not malicious), and 20 false negatives (not benign).

1. XAI for discovering spurious correlations. It is evident from the SHAP summary plot (Figure 6) that the GBM exhibits a strong reliance on the destination port,

7. XAI pipeline: https://github.com/tudelft-cda-lab/xai-pipeline

8. We recognize that the tutorial discusses a simple case study and that the features may have more complex relationships in reality. However, even this simple case occurs frequently in practice, as shown in [48].

9. LEMNA is excluded from the analysis since it provides remarkably fewer insights for class distinction compared to SHAP and LIME.

Feature	SHAP Value
State = 54	0.2339
SrcBytes = 186	-0.1756
StartTime = 1313593252	-0.1210
Dport = 80	0.1121
Sport = 1703	-0.1113
dTos = 0	0.0465
TotPkts = 8	0.0408
TotBytes = 492	-0.0295
Proto = 0	0.0266
Dur = 8.96	-0.0247
Dir = 2	0.0
sTos = 0	0.0

Feature	Value	LIME Rule	Weight
Sport	1703	Sport=1703	0.17
StartTime	1313593252	1313537772.00 < Start...	0.14
Dport	80	Dport = 80	0.12
TotPkts	8	TotPkts > 4.00	0.05
Proto	0	Proto=0	0.03
TotBytes	492	271.50 < TotBytes <= 4...	0.02
State	54	State=54	0.01
Dir	2	Dir = 2	0.01
Dur	8.96	0.13 < Dur <= 9.01	0.01
SrcBytes	186	83.50 < SrcBytes <= 1...	0.0

Feature	SHAP Value
Dport = 3389	0.5387
State = 16	0.0495
Sport = 4505	-0.0488
StartTime = 1313571534	-0.0475
dTos = 0	0.0274
TotPkts = 10	0.0209
TotBytes = 1076	-0.0077
SrcBytes = 437	-0.0057
Dur = 60.95	0.0032
Proto = 0	0.0
Dir = 2	0.0
sTos = 0	0.0

Figure 4. (Left): SHAP explanation for a false positive Netflow. (Middle): LIME explanation for a false positive Netflow. (Right): SHAP explanation for a false negative Netflow. Orange rows contribute positively, and blue rows contribute negatively towards the malicious label.

source port, and start time features. We also see this trend in the interpretable decision tree in Figure 3.

The reliance on the start time and source port features is problematic: start time is problematic because it represents Unix time, so each new Netflow will have a vastly different feature value compared to the ones seen in the training data, negatively impacting the test accuracy and the model's generalizability. For instance, a benign Netflow that the GBM considers as malicious with a probability of 65% suddenly becomes benign with a probability of 94% if we artificially perturb the start time to four weeks earlier. This implies that the model learns to predict *when a Netflow is generated*, rather than the Netflow's maliciousness.

Source port is problematic because it typically gets arbitrarily assigned by the operating system, and as such should not be indicative of malicious behaviour. However, the CTU-13 dataset uses only a small subset of VM-related port numbers [29], which inadvertently becomes indicative of malicious behaviour. This is a common shortcoming of lab-collected datasets [16]. Thus, it can also be considered an artefact of the experimental data.

It is noteworthy that start time and source port are perfectly valid features if the test set comes from CTU-13. Since we cannot expect real data to follow the same patterns as CTU-13, we consider them spurious features. This type of analysis is not common practice in the security literature: several recent and relatively popular works utilize the identified faulty features, see *e.g.,* [73], [101], [110]. Since these features are tightly coupled with the prediction label, standard feature selection methods are unlikely to get rid of them. This is where XAI can help.

The next logical step is to retrain the model without the spurious features. Doing so lowers the balanced accuracy of the GBM and decision tree to 74.4% and 58.1%, respectively. We argue that this is an improvement since the faulty features were making the classifier *appear performant* without being able to generalize in practice. Because cyber data is often noisy, sole reliance on performance metrics is generally meaningless, especially when spurious features are involved. Therefore, we recommend that like ablation studies, the identification and removal of spurious features should become a fundamental step in the design of ML pipelines.

2. XAI for finding causes of misclassifications. We find that post-hoc explanations must be supplemented with input data statistics to make meaningful inferences regarding the causes of misclassifications. For instance, we analyze a randomly sampled false positive Netflow. The local SHAP explanation (see Figure 4a) shows a heavy reliance on the state value of 54 and source bytes of 186. This information in itself is likely insufficient for an analyst to understand why the model made this mistake. However, combining this information with an analysis of the training data reveals that these feature values appear almost exclusively in malicious samples, thus identifying the cause of the false positive.

In another example, we analyze a randomly sampled false negative Netflow. The local SHAP explanation (see Figure 4c) shows a substantial reliance on the destination port 3389, which is associated with the remote desktop protocol (RDP). Internet-facing RDP servers commonly fall victim to cyber attacks[10], making it a likely indicator of suspicious activity. Yet strangely, the port 3389 has contributed heavily towards the opposite. Analyzing the training data reveals that RDP is mostly used by benign hosts in the CTU-13 dataset, due to which the model incorrectly classifies a malicious Netflow as benign. These examples reveal what appear to be sampling and confounding biases in the CTU-13 dataset.

> **Takeaway 7:** *Feature importance explanations do not provide the full picture in isolation. Instead, actionable insights can be obtained by combining the input data together with post-hoc explanations.*

3. Utility of different XAI types. All explanations are not created equal. Since XAI is meant to explain the behaviour of a model, testing the predictability of the model on a new (previously unseen) data instance, given a few explanations provides a simple estimate of the explanation's utility. In this sense, there is a clear divide between interpretable models and post-hoc explanations.

For a given interpretable model, such as the decision tree in Figure 3, it is almost trivial to predict how a new instance will be classified by following the decision path. However, since post-hoc explanations are mere approximations of the black-box model, it is difficult to predict how the GBM would classify a new instance, given its LIME and SHAP explanations. For instance, the local SHAP explanations provide feature importance with equality relationships (*e.g.,* see Figure 4a), which makes it impossible to predict how a new instance will be classified, even if it resembles the instances for which explanations are already available. This is because the explanations do not reveal the impact of slight feature perturbations on the classification. We encountered almost the same problem for LIME even though it considers a local neighbourhood to prevent this very issue.

10. http://darktrace.com/botnet-malware-remote-desktop-protocol-rdp

Moreover, post-hoc explainers compute their local neighbourhoods differently, causing explanations for the same model prediction to differ. Going back to the false positive example, SHAP (Figure 4a) heavily relies on the state feature, while LIME (Figure 4b) assigns very low importance to it. Also, while SHAP considers dTos to be important, it does not even appear in LIME. This disagreement problem between feature attribution methods has recently been discussed by Krishna *et al.* [77]. Based on their metric, there is a 25.5% disagreement rate between the top-3 features of SHAP and LIME for our 140 Netflows. This exemplifies the mismatch between the black-box model and its explanations and makes a strong case for learning interpretable models from the get-go.

Furthermore, the correct interpretation of post-hoc explanations often relies on how well the explainee understands the underlying mechanisms of the method, reiterating the importance of user studies in explanation evaluation. For instance, while both local-SHAP and LIME show feature importance, their explanation interpretation can be very different. We found that LIME assigns very low weights to all features for almost all the Netflows. This does not imply that similar Netflows should have the same label, as one would intuitively expect, but rather that LIME has low confidence about the prediction given its local surroundings. Thus, an unsuspecting analyst might draw misleading conclusions by overly relying on intuition rather than the understanding of the method [82].

Takeaway 8: *LIME and local-SHAP are both feature attribution methods but their interpretations can be different. Working knowledge of post-hoc explainers is cardinal for correctly interpreting the explanations.*

9. Discussion and Open Problems

Below, we identify open problems and provide recommendations for further XAI research within cybersecurity:

User study crisis. The lack of qualitative validation among decision support papers is alarming. While *model users* are the most common consumers of explanations in the security literature, they have regularly been excluded from the evaluation process *(Takeaways 1-3, 8)*. The evaluation of robustness and fidelity does not guarantee usability, which is arguably an equally important trait of good explanations. However, usability is rarely taken into account when designing evaluation criteria for effective security explanations, see *e.g.,* [52], [129]. Since analyst time is expensive, it may be beneficial to develop proxy tasks and metrics on which to evaluate new research instead. A handful of studies have incorporated human cognition in their metric definition. For example, Islam *et al.* [71] quantify the complexity of post-hoc explanations in terms of cognitive chunks, and Dolejš *et al.* [43] quantify the added opaqueness of explanations *w.r.t.* known interpretable models. Alternatively, in the absence of security practitioners, newly developed tools could be peer-reviewed regarding their usability, *e.g.,* during conference artefact evaluation sessions. Furthermore, disentangling and specifying stakeholders should also provide clarity regarding the intended subjects for user studies.

Robustness *vs.* interpretability. The role of *model designers* is minimized in the security literature with merely 22.3% of the literature focused on model & explainer verification *(Takeaways 4-5, 7-8)*. Since trust manifests inherently differently in the security domain, specialized XAI methods are needed to bolster practitioner trust in ML pipelines. While the tutorial in §8 helps model designers get started with XAI-enabled model verification, the role of XAI in tamper-resistant feature selection and robust model learning remains unclear. Another related question is regarding the relationship between robustness and interpretability: initial research eludes to robust models being more interpretable than non-robust models [114], requiring further research in this direction.

Price of interpretability. If interpretable surrogate models are to be used for model verification, they must be certifiably equivalent to their black-box parent models for the evaluation to be meaningful. In this sense, directly learning a robust interpretable model (as opposed to learning a black-box model explained by a surrogate) may prove more helpful in establishing trust. Yet, only 25.9% of the studies we reviewed adopt interpretable models, while the majority of them focus on applying post-hoc explainability. The discussion regarding the *'price of interpretability'* (measuring the trade-off between explainability and performance) [20] requires special considerations in cybersecurity. We believe that the presence of an adversary and the prevalence of spurious features will likely make this trade-off less pronounced compared to other fields. However, further research is warranted in this area. Furthermore, it may be possible to exploit the power of post-hoc explanations without losing interpretability: black-box models may be used as a benchmark to guide the search for better interpretable models. Post-hoc explanations can provide actionable intelligence regarding features and parameters for interpretable models. In this way, we view post-hoc explainers and interpretable models as complementary methods rather than as alternatives.

Privacy-preserving explanations. In addition to attacking the XAI module, adversaries can also utilize explanations, much like *model users*, but with a devious intent *(Takeaway 6)*. This makes it difficult to provide explanations to model designers and model users without the adversaries also taking advantage of them. There is some preliminary work that studies the trade-off between explainability and privacy in order to select privacy-preserving explanations [23]. However, such explanations could still be used to bolster attacks on model integrity and availability. Therefore, this is also an urgent avenue for future research.

10. Conclusions

We systematize available research that utilizes explainable models for solving security problems. We identify 3 cybersecurity stakeholders that employ XAI for 4 research objectives within a typical ML pipeline. Distilled from a diverse body of literature, this overview streamlines existing research on explainability within cybersecurity and provides a starting point for practitioners.

We found evidence that the security literature does not always disentangle model users and designers. In addition, only 22.3% of the security literature focuses on model & explanation verification. This is problematic because

model designers have a critical role in ensuring the correctness and security of an ML pipeline. With regards to model correctness, we specifically provide a walk-through tutorial of how model designers can successfully detect and discard spurious features using SHAP & LIME. At the same time, the example also exposed the disagreement problem between local explanations and showed that SHAP & LIME have different interpretations. Thus, model designers must have a working knowledge of the explanation method in order to draw correct conclusions.

Moreover, adversaries can not only attack the XAI component but can also utilize explanations to compromise the confidentiality, integrity and availability of a model. Meanwhile, research on limiting these abuses is almost non-existent. Finally, the lack of user validation in XAI-enabled user assistance, and the lack of interpretability by design shows the substantial margins of improvement within the field of XAI for cybersecurity.

Acknowledgements. We thank Daniël Meinsma for his contributions to the literature review, and the anonymous reviewers for their valuable feedback. This work was made possible by TTW VIDI project 17541 (LIMIT) and EU H2020 project 952647 (AssureMOSS).

References

[1] Abel A. Reyes, Francisco D. Vaca, Gabriel A. Castro Aguayo, Quamar Niyaz, and Vijay Devabhaktuni. A Machine Learning Based Two-Stage Wi-Fi Network Intrusion Detection System. *Electronics*, 9(10), 2020.

[2] Amina Adadi and Mohammed Berrada. Peeking inside the black-box: a survey on explainable artificial intelligence (XAI). *IEEE Access*, 6:52138–52160, 2018.

[3] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. Sanity checks for saliency maps. *Advances in neural information processing systems*, 31, 2018.

[4] Seyoung Ahn, Jeehyeong Kim, Soo Young Park, and Sunghyun Cho. Explaining deep learning-based traffic classification using a genetic algorithm. *IEEE Access*, 9:4738–4751, 2020.

[5] Ulrich Aivodji, Hiromi Arai, Olivier Fortineau, Sébastien Gambs, Satoshi Hara, and Alain Tapp. Fairwashing: the risk of rationalization. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 161–170. PMLR, 09–15 Jun 2019.

[6] Sefi Akerman, Edan Habler, and Asaf Shabtai. VizADS-B: Analyzing sequences of ADS-B images using explainable convolutional LSTM encoder-decoder to detect cyber attacks. *arXiv preprint arXiv:1906.07921*, 2019.

[7] Kenneth B Alperin, Allan B Wollaber, and Steven R Gomez. Improving Interpretability for Cyber Vulnerability Assessment Using Focus and Context Visualizations. In *IEEE Symposium on Visualization for Cyber Security (VizSec)*, pages 30–39. IEEE, 2020.

[8] Mohammed Alshiekh, Roderick Bloem, Rüdiger Ehlers, Bettina Könighofer, Scott Niekum, and Ufuk Topcu. Safe reinforcement learning via shielding. In *AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[9] David Alvarez-Melis and Tommi S Jaakkola. On the robustness of interpretability methods. *arXiv preprint arXiv:1806.08049*, 2018.

[10] Marco Angelini, Leonardo Aniello, Simone Lenti, Giuseppe Santucci, and Daniele Ucci. The goods, the bads and the uglies: Supporting decisions in malware detection through visual analytics. In *IEEE Symposium on Visualization for Cyber Security (VizSec)*, pages 1–8. IEEE, 2017.

[11] Sule Anjomshoae, Amro Najjar, Davide Calvaresi, and Kary Främling. Explainable agents and robots: Results from a systematic literature review. In *AAMAS*, pages 1078–1088. International Foundation for Autonomous Agents and Multiagent Systems, 2019.

[12] Liat Antwarg, Ronnie Mindlin Miller, Bracha Shapira, and Lior Rokach. Explaining anomalies detected by autoencoders using Shapley Additive Explanations. *Expert Systems with Applications*, 186:115736, 2021.

[13] Giovanni Apruzzese, Pavel Laskov, Edgardo Montes de Oca, Wissam Mallouli, Luis Burdalo Rapa, Athanasios Vasileios Grammatopoulos, and Fabio Di Franco. The role of machine learning in cybersecurity. *Digital Threats: Research and Practice*, 2022.

[14] Carmelo Ardito, Yashar Deldjoo, Eugenio Di Sciascio, and Fatemeh Nazary. Revisiting security threat on smart grids: accurate and interpretable fault location prediction and type classification. In *Italian Conference on CyberSecurity*, 2021.

[15] Carmelo Ardito, Tommaso Di Noia, Eugenio Di Sciascio, Domenico Lofù, Andrea Pazienza, and Felice Vitulano. An artificial intelligence cyberattack detection system to improve threat reaction in e-health. In *Italian Conference on Cybersecurity*, 2021.

[16] Daniel Arp, Erwin Quiring, Feargus Pendlebury, Alexander Warnecke, Fabio Pierazzi, Christian Wressnegger, Lorenzo Cavallaro, and Konrad Rieck. Dos and don'ts of machine learning in computer security. In *Proc. of the USENIX Security Symposium*, 2022.

[17] Daniel Arp, Michael Spreitzenbarth, Malte Hubner, Hugo Gascon, Konrad Rieck, and CERT Siemens. Drebin: Effective and explainable detection of android malware in your pocket. In *Proceedings of NDSS*, volume 14, pages 23–26, 2014.

[18] Franziska Becker, Arthur Drichel, Christoph Müller, and Thomas Ertl. Interpretable Visualizations of Deep Neural Networks for Domain Generation Algorithm Detection. In *IEEE Symposium on Visualization for Cyber Security (VizSec)*, pages 25–29. IEEE, 2020.

[19] Nourhène Ben Rabah, Bénédicte Le Grand, and Manuele Kirsch Pinheiro. IoT Botnet Detection using Black-box Machine Learning Models: the Trade-off between Performance and Interpretability. In *IEEE International Conference on Enabling Technologies*, pages 101–106, 2021.

[20] Dimitris Bertsimas, Arthur Delarue, Patrick Jaillet, and Sebastien Martin. The price of interpretability. *arXiv preprint arXiv:1907.03419*, 2019.

[21] Battista Biggio, Igino Corona, Davide Maiorca, Blaine Nelson, Nedim Šrndić, Pavel Laskov, Giorgio Giacinto, and Fabio Roli. Evasion attacks against machine learning at test time. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 387–402. Springer, 2013.

[22] Battista Biggio and Fabio Roli. Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognition*, 84:317–331, 2018.

[23] Sumitra Biswal. System Preserving Explainability and Confidentiality (SPEC). In *4th workshop on machine learning for cybersecurity*. Springer, 2022.

[24] Mathias Blumreiter, Joel Greenyer, Francisco Javier Chiyah Garcia, Verena Klös, Maike Schwammberger, Christoph Sommer, Andreas Vogelsang, and Andreas Wortmann. Towards self-explainable cyber-physical systems. In *ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion*, pages 543–548. IEEE, 2019.

[25] Virginia Braun and Victoria Clarke. Using thematic analysis in psychology. *Qualitative research in psychology*, 3(2):77–101, 2006.

[26] Andy Brown, Aaron Tuor, Brian Hutchinson, and Nicole Nichols. Recurrent neural network attention mechanisms for interpretable system log anomaly detection. In *Workshop on Machine Learning for Computing Systems*, pages 1–8, 2018.

[27] Stefano Calzavara, Claudio Lucchese, Gabriele Tolomei, Seyum Assefa Abebe, and Salvatore Orlando. Treant: training evasion-aware decision trees. *Data Mining and Knowledge Discovery*, 34(5):1390–1420, 2020.

[28] Clinton Cao, Agathe Blaise, Sicco Verwer, and Filippo Rebecchi. Learning State Machines to Monitor and Detect Anomalies on a Kubernetes Cluster. In *International Conference on Availability, Reliability and Security*, 2022.

[29] Clinton Cao, Annibale Panichella, Sicco Verwer, Agathe Blaise, and Filippo Rebecchi. Encoding NetFlows for State-Machine Learning. *arXiv preprint arXiv:2207.03890*, 2022.

[30] Fran Casino, Thomas K. Dasaklis, Georgios P. Spathoulas, Marios Anagnostopoulos, Amrita Ghosal, István Bořočz, Agusti Solanas, Mauro Conti, and Constantinos Patsakis. Research Trends, Challenges, and Emerging Topics in Digital Forensics: A Review of Reviews. *IEEE Access*, 10:25464–25493, 2022.

[31] Yidong Chai, Yonghang Zhou, Weifeng Li, and Yuanchun Jiang. An explainable multi-modal hierarchical attention model for developing phishing threat intelligence. *IEEE Transactions on Dependable and Secure Computing*, 19(2):790–803, 2021.

[32] Saikat Chakraborty, Rahul Krishna, Yangruibo Ding, and Baishakhi Ray. Deep Learning based Vulnerability Detection: Are We There Yet. *IEEE Transactions on Software Engineering*, PP:1–1, 06 2021.

[33] Hongge Chen, Huan Zhang, Duane Boning, and Cho-Jui Hsieh. Robust decision trees against adversarial examples. In *International Conference on Machine Learning*, pages 1122–1131. PMLR, 2019.

[34] Yanjiao Chen, Yijie Bai, Richard Mitev, Kaibo Wang, Ahmad-Reza Sadeghi, and Wenyuan Xu. FakeWake: Understanding and Mitigating Fake Wake-up Words of Voice Assistants. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, CCS '21, page 1861–1883, New York, NY, USA, 2021. Association for Computing Machinery.

[35] Chia Yuan Cho, Domagoj Babi ć, Eui Chul Richard Shin, and Dawn Song. Inference and analysis of formal models of botnet command and control protocols. In *ACM conference on Computer and communications security*, pages 426–439, 2010.

[36] Zheng Leong Chua, Shiqi Shen, Prateek Saxena, and Zhenkai Liang. Neural nets can learn function type signatures from binaries. In *Proc. of the USENIX Security Symposium*, pages 99–116, 2017.

[37] James Clear. *Atomic habits: An easy & proven way to build good habits & break bad ones*. Penguin, 2018.

[38] Paolo Milani Comparetti, Gilbert Wondracek, Christopher Kruegel, and Engin Kirda. Prospex: Protocol specification extraction. In *IEEE symposium on security and privacy*, pages 110–125. IEEE, 2009.

[39] Igino Corona, Battista Biggio, Matteo Contini, Luca Piras, Roberto Corda, Mauro Mereu, Guido Mureddu, Davide Ariu, and Fabio Roli. Deltaphish: Detecting phishing webpages in compromised websites. In *European Symposium on Research in Computer Security*, pages 370–388. Springer, 2017.

[40] Weidong Cui, Jayanthkumar Kannan, and Helen J Wang. Discoverer: Automatic Protocol Reverse Engineering from Network Traces. In *Proc. of the USENIX Security Symposium*, pages 1–14, 2007.

[41] Kim de Bie, Ana Lucic, and Hinda Haned. To Trust or Not to Trust a Regressor: Estimating and Explaining Trustworthiness of Regression Predictions. *arXiv preprint arXiv:2104.06982*, 2021.

[42] Luca Demetrio, Battista Biggio, Giovanni Lagorio, Fabio Roli, and Armando Alessandro. Explaining vulnerabilities of deep learning to adversarial malware binaries. In *Italian Conference on Cyber Security*, volume 2315, 2019.

[43] Jan Dolejš and Martin Jureček. Interpretability of Machine Learning-Based Results of Malware Detection Using a Set of Rules. In *Cybersecurity for Artificial Intelligence*, pages 107–136. Springer, 2022.

[44] Ann-Kathrin Dombrowski, Maximillian Alber, Christopher Anders, Marcel Ackermann, Klaus-Robert Müller, and Pan Kessel. Explanations can be manipulated and geometry is to blame. *Advances in Neural Information Processing Systems*, 32, 2019.

[45] Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017.

[46] Xu Duan, Jingzheng Wu, Shouling Ji, Zhiqing Rui, Tianyue Luo, Mutian Yang, and Yanjun Wu. VulSniper: Focus Your Attention to Shoot Fine-Grained Vulnerabilities. In *Proceedings of IJCAI*, pages 4665–4671, 08 2019.

[47] Gunning D Aha DW. DARPA's explainable artificial intelligence program. *AI Mag*, 40(2):44, 2019.

[48] Laurens D'hooge, Miel Verkerken, Bruno Volckaert, Tim Wauters, and Filip De Turck. Establishing the contaminating effect of metadata feature inclusion in machine-learned network intrusion detection models. In *Detection of Intrusions and Malware, and Vulnerability Assessment: 19th International Conference, DIMVA 2022, Cagliari, Italy, June 29–July 1, 2022, Proceedings*, pages 23–41. Springer, 2022.

[49] Paul Fiterau-Brostean, Bengt Jonsson, Robert Merget, Joeri De Ruiter, Konstantinos Sagonas, and Juraj Somorovsky. Analysis of DTLS Implementations Using Protocol State Fuzzing. In *Proc. of the USENIX Security Symposium*, pages 2523–2540, 2020.

[50] Hidde Fokkema, Rianne de Heide, and Tim van Erven. Attribution-based Explanations that Provide Recourse Cannot be Robust. *arXiv preprint arXiv:2205.15834*, 2022.

[51] Antonio Galli, Stefano Marrone, Vincenzo Moscato, and Carlo Sansone. Reliability of explainable artificial intelligence in adversarial perturbation scenarios. In *International Conference on Pattern Recognition*, pages 243–256. Springer, 2021.

[52] Tom Ganz, Martin Härterich, Alexander Warnecke, and Konrad Rieck. Explaining Graph Neural Networks for Vulnerability Discovery. In *ACM Workshop on Artificial Intelligence and Security*, pages 145–156, 2021.

[53] S. García, M. Grill, J. Stiborek, and A. Zunino. An empirical comparison of botnet detection methods. *Computers & Security*, 45:100–123, 2014.

[54] Timon Gehr, Matthew Mirman, Dana Drachsler-Cohen, Petar Tsankov, Swarat Chaudhuri, and Martin Vechev. AI2: Safety and robustness certification of neural networks with abstract interpretation. In *IEEE symposium on security and privacy*, pages 3–18. IEEE, 2018.

[55] Marzyeh Ghassemi, Luke Oakden-Rayner, and Andrew L Beam. The false hope of current approaches to explainable artificial intelligence in health care. *The Lancet Digital Health*, 3(11):e745–e750, 2021.

[56] Amirata Ghorbani, Abubakar Abid, and James Zou. Interpretation of neural networks is fragile. In *AAAI conference on artificial intelligence*, volume 33, pages 3681–3688, 2019.

[57] Paolo Giudici and Emanuela Raffinetti. Explainable AI methods in cyber risk management. *Quality and Reliability Engineering International*, 2021.

[58] Bryce Goodman and Seth Flaxman. European Union regulations on algorithmic decision-making and a "right to explanation". *AI magazine*, 38(3):50–57, 2017.

[59] Berk Gulmezoglu. XAI-based Microarchitectural Side-Channel Analysis for Website Fingerprinting Attacks and Defenses. *IEEE Transactions on Dependable and Secure Computing*, pages 1–1, 2021.

[60] Wenbo Guo, Dongliang Mu, Jun Xu, Purui Su, Gang Wang, and Xinyu Xing. LEMNA: Explaining Deep Learning Based Security Applications. In *ACM SIGSAC Conference on Computer and Communications Security*, CCS '18, page 364–379, New York, NY, USA, 2018. Association for Computing Machinery.

[61] Britta Hale, Douglas L Van Bossuyt, Nikolaos Papakonstantinou, and Bryan O'Halloran. A zero-trust methodology for security of complex systems with machine learning components. In *International design engineering technical conferences and computers and information in engineering conference*, volume 85376. American Society of Mechanical Engineers, 2021.

[62] Patrick Hall, Navdeep Gill, and Nicholas Schmidt. Proposed Guidelines for the Responsible Use of Explainable Machine Learning, 2019.

[63] Dongqi Han, Zhiliang Wang, Wenqi Chen, Ying Zhong, Su Wang, Han Zhang, Jiahai Yang, Xingang Shi, and Xia Yin. DeepAID: interpreting and improving deep learning-based anomaly detection in security applications. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 3197–3217, 2021.

[64] Lars Kai Hansen and Laura Rieger. Interpretability in intelligent systems–a new concept? In *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning*, pages 41–49. Springer, 2019.

[65] Swetha Hariharan, Anusha Velicheti, AS Anagha, Ciza Thomas, and N Balakrishnan. Explainable Artificial Intelligence in Cybersecurity: A Brief Review. In *International Conference on Security and Privacy*, pages 1–12. IEEE, 2021.

[66] Jamie Hayes, Borja Balle, and M Pawan Kumar. Learning to be adversarially robust and differentially private. *arXiv preprint arXiv:2201.02265*, 2022.

[67] Eric Holder and Ning Wang. Explainable artificial intelligence (XAI) interactively working with humans as a junior cyber analyst. *Human-Intelligent Systems Integration*, pages 1–15, 2021.

[68] Xiaowei Huang, Daniel Kroening, Wenjie Ruan, James Sharp, Youcheng Sun, Emese Thamo, Min Wu, and Xinping Yi. A survey of safety and trustworthiness of deep neural networks: Verification, testing, adversarial attack and defence, and interpretability. *Computer Science Review*, 37:100270, 2020.

[69] Chanwoong Hwang and Taejin Lee. E-SFD: Explainable Sensor Fault Detection in the ICS Anomaly Detection System. *IEEE Access*, 9:140470–140486, 2021.

[70] Giacomo Iadarola, Fabio Martinelli, Francesco Mercaldo, and Antonella Santone. Towards an interpretable deep learning model for mobile malware detection and family identification. *Computers & Security*, 105:102198, 2021.

[71] Sheikh Rabiul Islam, William Eberle, and Sheikh K Ghafoor. Towards quantification of explainability in explainable artificial intelligence methods. In *The thirty-third international flairs conference*, 2020.

[72] Rupesh Raj Karn, Prabhakar Kudva, Hai Huang, Sahil Suneja, and Ibrahim M Elfadel. Cryptomining detection in container clouds using system calls and explainable machine learning. *IEEE Transactions on Parallel and Distributed Systems*, 32(3):674–691, 2020.

[73] Riaz Ullah Khan, Xiaosong Zhang, Rajesh Kumar, Abubakar Sharif, Noorbakhsh Amiri Golilarz, and Mamoun Alazab. An adaptive multi-layer botnet detection technique using machine learning classifiers. *Applied Sciences*, 9(11):2375, 2019.

[74] Martin Kinkead, Stuart Millar, Niall McLaughlin, and Philip O'Kane. Towards explainable CNNs for Android malware detection. *Procedia Computer Science*, 184:959–965, 2021.

[75] Anurag Koul, Sam Greydanus, and Alan Fern. Learning finite state representations of recurrent policy networks. *arXiv preprint arXiv:1811.12530*, 2018.

[76] Vasilios Koutsokostas, Nikolaos Lykousas, Theodoros Apostolopoulos, Gabriele Orazi, Amrita Ghosal, Fran Casino, Mauro Conti, and Constantinos Patsakis. Invoice #31415 attached: Automated analysis of malicious Microsoft Office documents. *Computers & Security*, 114:102582, 2022.

[77] Satyapriya Krishna, Tessa Han, Alex Gu, Javin Pombra, Shahin Jabbari, Steven Wu, and Himabindu Lakkaraju. The Disagreement Problem in Explainable Machine Learning: A Practitioner's Perspective. *arXiv preprint arXiv:2202.01602*, 2022.

[78] Aditya Kuppa and Nhien-An Le-Khac. Black box attacks on explainable artificial intelligence (XAI) methods in cyber security. In *IJCNN*, pages 1–8. IEEE, 2020.

[79] Aditya Kuppa and Nhien-An Le-Khac. Adversarial XAI methods in cybersecurity. *IEEE Transactions on Information Forensics and Security*, 16:4924–4938, 2021.

[80] Adarsh Kyadige, Ethan M. Rudd, and Konstantin Berlin. Learning from Context: A Multi-View Deep Learning Architecture for Malware Detection. In *IEEE Security and Privacy Workshops (SPW)*, pages 1–7, 2020.

[81] Triet Le, Huaming Chen, and Muhammad Ali Babar. A Survey on Data-driven Software Vulnerability Assessment and Prioritization. *ACM Computing Surveys*, 07 2021.

[82] Matthew L Leavitt and Ari Morcos. Towards falsifiable interpretability research. *arXiv preprint arXiv:2010.12016*, 2020.

[83] Huiling Li, Jun Wu, Hansong Xu, Gaolei Li, and Mohsen Guizani. Explainable Intelligence-Driven Defense Mechanism Against Advanced Persistent Threats: A Joint Edge Game and AI Approach. *IEEE Transactions on Dependable and Secure Computing*, 19(2):757–775, 2021.

[84] Yi Li, Shaohua Wang, and Nguyen Tien. Vulnerability detection with fine-grained interpretations. In *Proceedings of the 29th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 292–303, 08 2021.

[85] Qin Lin, Sridha Adepu, Sicco Verwer, and Aditya Mathur. TABOR: A graphical model-based approach for anomaly detection in industrial control systems. In *asia conference on computer and communications security*, pages 525–536, 2018.

[86] Yi-Shan Lin, Wen-Chuan Lee, and Z Berkay Celik. What do you see? Evaluation of explainable artificial intelligence (XAI) interpretability through neural backdoors. *arXiv preprint arXiv:2009.10639*, 2020.

[87] Yun Lin, Ruofan Liu, Dinil Mon Divakaran, Jun Yang Ng, Qing Zhou Chan, Yiwen Lu, Yuxuan Si, Fan Zhang, and Jin Song Dong. Phishpedia: a hybrid deep learning based approach to visually identify phishing webpages. In *Proc. of the USENIX Security Symposium*, pages 3793–3810, 2021.

[88] Zachary C Lipton. The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3):31–57, 2018.

[89] Hong Liu, Chen Zhong, Awny Alnusair, and Sheikh Rabiul Islam. FAIXID: a framework for enhancing AI explainability of intrusion detection results using data cleaning techniques. *Journal of Network and Systems Management*, 29(4):1–30, 2021.

[90] Daniel Lowd and Christopher Meek. Adversarial learning. In *ACM SIGKDD international conference on Knowledge discovery in data mining*, pages 641–647, 2005.

[91] Basim Mahbooba, Mohan Timilsina, Radhya Sahal, and Martin Serrano. Explainable artificial intelligence (XAI) to enhance trust management in intrusion detection systems using decision tree model. *Complexity*, 2021, 2021.

[92] Samaneh Mahdavifar and Ali A Ghorbani. DeNNeS: deep embedded neural network expert system for detecting cyber attacks. *Neural Computing and Applications*, 32(18):14753–14780, 2020.

[93] Daniel L Marino, Chathurika S Wickramasinghe, and Milos Manic. An adversarial approach for explainable AI in intrusion detection systems. In *Annual Conference of the IEEE Industrial Electronics Society*, pages 3237–3243. IEEE, 2018.

[94] Sherin Mary Mathews. Explainable artificial intelligence applications in NLP, biomedical, and malware classification: a literature review. In *Intelligent computing-proceedings of the computing conference*, pages 1269–1292. Springer, 2019.

[95] Tim Miller. Explanation in artificial intelligence: Insights from the social sciences. *Artificial intelligence*, 267:1–38, 2019.

[96] Azqa Nadeem, Christian Hammerschmidt, Carlos H Gañán, and Sicco Verwer. Beyond labeling: Using clustering to build network behavioral profiles of malware families. In *Malware Analysis Using Artificial Intelligence and Deep Learning*, pages 381–409. Springer, 2021.

[97] Azqa Nadeem, Vera Rimmer, Wouter Joosen, and Sicco Verwer. Intelligent Malware Defenses. In *Security and Artificial Intelligence*, pages 217–253. Springer, 2022.

[98] Azqa Nadeem, Sicco Verwer, Stephen Moskal, and Shanchieh Jay Yang. Alert-driven Attack Graph Generation using S-PDFA. *IEEE Transactions on Dependable and Secure Computing*, 2021.

[99] Azqa Nadeem, Shanchieh Jay Yang, and Sicco Verwer. Learning about the adversary. In *Autonomous intelligent agents for cyber defense*. Springer, 2023.

[100] Claire Nicodeme. Build confidence and acceptance of AI-based decision support systems-Explainable and liable AI. In *International Conference on Human System Interaction*, pages 20–23. IEEE, 2020.

[101] Miguel Nicolau, James McDermott, et al. Learning neural representations for network anomaly detection. *IEEE transactions on cybernetics*, 49(8):3074–3087, 2018.

[102] Andrew P Norton and Yanjun Qi. Adversarial-Playground: A visualization suite showing how adversarial examples fool deep learning. In *IEEE symposium on visualization for cyber security (VizSec)*, pages 1–4. IEEE, 2017.

[103] Megan Nyre-Yu, Elizabeth Susan Morris, Blake Cameron Moss, Charles Smutz, and Michael Smith. Explainable AI in Cybersecurity Operations: Lessons Learned from XAI Tool Deployment. In *Usable Security and Privacy (USEC) Symposium*, 2022.

[104] Lace M Padilla, Sarah H Creem-Regehr, Mary Hegarty, and Jeanine K Stefanucci. Decision making with visualizations: a cognitive framework across disciplines. *Cognitive research: principles and implications*, 3(1):1–25, 2018.

[105] Cecilia Panigutti, Andrea Beretta, Fosca Giannotti, and Dino Pedreschi. Understanding the impact of explanations on advice-taking: a user study for AI-based clinical Decision Support Systems. In *CHI Conference on Human Factors in Computing Systems*, pages 1–9, 2022.

[106] Nicolas Papernot, Patrick McDaniel, Arunesh Sinha, and Michael P Wellman. SoK: Security and privacy in machine learning. In *IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 399–414. IEEE, 2018.

[107] Jose N Paredes, Juan Carlos L Teze, Gerardo I Simari, and Maria Vanina Martinez. On the Importance of Domain-specific Explanations in AI-based Cybersecurity Systems (Technical Report). *arXiv preprint arXiv:2108.02006*, 2021.

[108] Gonzalo De La Torre Parra, Luis Selvera, Joseph Khoury, Hector Irizarry, Elias Bou-Harb, and Paul Rad. Interpretable Federated Transformer Log Learning for Cloud Threat Forensics. In *Proceedings of NDSS*, 2022.

[109] Charles P Pfleeger and Shari Lawrence Pfleeger. *Analyzing computer security: A threat/vulnerability/countermeasure approach*. Prentice Hall Professional, 2012.

[110] Rojalina Priyadarshini and Rabindra Kumar Barik. A deep learning based intelligent framework to mitigate DDoS attack in fog environment. *Journal of King Saud University-Computer and Information Sciences*, 2019.

[111] Konrad Rieck, Thorsten Holz, Carsten Willems, Patrick Düssel, and Pavel Laskov. Learning and classification of malware behavior. In *DIMVA*, pages 108–125. Springer, 2008.

[112] Ribana Roscher, Bastian Bohn, Marco F Duarte, and Jochen Garcke. Explainable machine learning for scientific insights and discoveries. *IEEE Access*, 8:42200–42216, 2020.

[113] Ishai Rosenberg, Asaf Shabtai, Yuval Elovici, and Lior Rokach. Adversarial machine learning attacks and defense methods in the cyber security domain. *ACM Computing Surveys (CSUR)*, 54(5):1–36, 2021.

[114] Andrew Ross and Finale Doshi-Velez. Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. In *AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[115] Rebecca Russell, Louis Kim, Lei Hamilton, Tomo Lazovich, Jacob Harer, Onur Ozdemir, Paul Ellingwood, and Marc McConley. Automated Vulnerability Detection in Source Code Using Deep Representation Learning. In *17th IEEE international conference on machine learning and applications (ICMLA)*, pages 757–762, 12 2018.

[116] Laasya Samhita and Hans J Gross. The "Clever Hans Phenomenon" revisited. *Communicative & integrative biology*, 6(6):e27122, 2013.

[117] Sagar Samtani, Murat Kantarcioglu, and Hsinchun Chen. Trailblazing the artificial intelligence for cybersecurity discipline: a multi-disciplinary research roadmap, 2020.

[118] Giorgio Severi, Jim Meyer, Scott Coull, and Alina Oprea. Explanation-Guided Backdoor Poisoning Attacks Against Malware Classifiers. In *Proc. of the USENIX Security Symposium*, pages 1487–1504, 2021.

[119] Reza Shokri, Martin Strobel, and Yair Zick. On the privacy risks of model explanations. In *AAAI/ACM Conference on AI, Ethics, and Society*, pages 231–241, 2021.

[120] Awalin Sopan, Matthew Berninger, Murali Mulakaluri, and Raj Katakam. Building a Machine Learning Model for the SOC, by the Input from the SOC, and Analyzing it for the SOC. In *IEEE Symposium on Visualization for Cyber Security (VizSec)*, pages 1–8. IEEE, 2018.

[121] Alexey Surkov, Val Srinivas, and Jill Gregorie. Unleashing the power of machine learning models in banking through Explainable Artificial Intelligence (XAI), Jun 2022.

[122] Mateusz Szczepański, Michał Choraś, Marek Pawlicki, and Rafał Kozik. Achieving explainability of intrusion detection system by hybrid oracle-explainer approach. In *IJCNN*, pages 1–8. IEEE, 2020.

[123] Thijs van Ede, Hojjat Aghakhani, Noah Spahn, Riccardo Bortolameotti, Marco Cova, Andrea Continella, Maarten van Steen, Andreas Peter, Christopher Kruegel, and Giovanni Vigna. DeepCASE: Semi-Supervised Contextual Analysis of Security Events". In *IEEE Symposium on Security and Privacy*, 2022.

[124] Michael Van Lent, William Fisher, and Michael Mancuso. An explainable artificial intelligence system for small-unit tactical behavior. In *Proceedings of the national conference on artificial intelligence*, pages 900–907. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2004.

[125] Luca Vigano and Daniele Magazzeni. Explainable security. In *IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*, pages 293–300. IEEE, 2020.

[126] Daniël Vos and Sicco Verwer. Efficient training of robust decision trees against adversarial examples. In *International Conference on Machine Learning*, pages 10586–10595. PMLR, 2021.

[127] Maonan Wang, Kangfeng Zheng, Yanqing Yang, and Xiujuan Wang. An Explainable Machine Learning Framework for Intrusion Detection Systems. *IEEE Access*, 8:73127–73141, 2020.

[128] Shanshan Wang, Zhenxiang Chen, Lei Zhang, Qiben Yan, Bo Yang, Lizhi Peng, and Zhongtian Jia. Trafficav: An effective and explainable detection of mobile malware behavior using network traffic. In *IEEE/ACM International Symposium on Quality of Service*, pages 1–6. IEEE, 2016.

[129] Alexander Warnecke, Daniel Arp, Christian Wressnegger, and Konrad Rieck. Evaluating Explanation Methods for Deep Learning in Security. In *IEEE european symposium on security and privacy (EuroS&P)*, pages 158–174, 09 2020.

[130] Gail Weiss, Yoav Goldberg, and Eran Yahav. Extracting automata from recurrent neural networks using queries and counterexamples. In *International Conference on Machine Learning*, pages 5247–5256. PMLR, 2018.

[131] Chathurika S Wickramasinghe, Kasun Amarasinghe, Daniel L Marino, Craig Rieger, and Milos Manic. Explainable unsupervised machine learning for cyber-physical systems. *IEEE Access*, 9:131824–131843, 2021.

[132] Anjana Wijekoon and Nirmalie Wiratunga. Reasoning with counterfactual explanations for code vulnerability detection and correction. In *AI-Cybersec@ SGAI*, pages 1–13, 2021.

[133] Bozhi Wu, Sen Chen, Cuiyun Gao, Lingling Fan, Yang Liu, Weiping Wen, and Michael R Lyu. Why an android app is classified as malware: Toward malware classification interpretation. *ACM TOSEM*, 30(2):1–29, 2021.

[134] Xian Wu, Wenbo Guo, Hua Wei, and Xinyu Xing. Adversarial policy training against deep reinforcement learning. In *Proc. of the USENIX Security Symposium*, pages 1883–1900, 2021.

[135] Weiming Xiang, Patrick Musau, Ayana A Wild, Diego Manzanas Lopez, Nathaniel Hamilton, Xiaodong Yang, Joel Rosenfeld, and Taylor T Johnson. Verification for machine learning, autonomy, and neural networks survey. *arXiv preprint arXiv:1810.01989*, 2018.

[136] Jing Xu, Minhui Xue, and Stjepan Picek. Explainability-based backdoor attacks against graph neural networks. In *ACM Workshop on Wireless Security and Machine Learning*, pages 31–36, 2021.

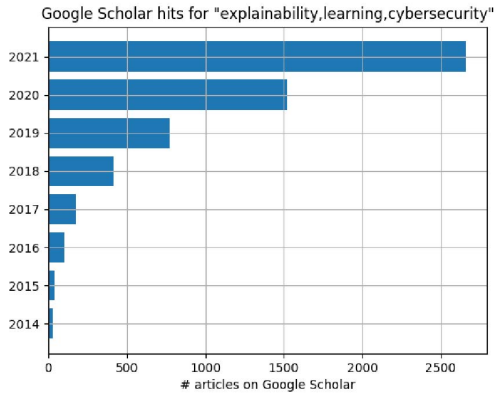

Figure 5. Prevalence of XAI literature from 2014-2021.

[137] Fabian Yamaguchi, Alwin Maier, Hugo Gascon, and Konrad Rieck. Automatic inference of search patterns for taint-style vulnerabilities. In *IEEE symposium on security and privacy*, pages 797–812. IEEE, 2015.

[138] Limin Yang, Wenbo Guo, Qingying Hao, Arridhana Ciptadi, Ali Ahmadzadeh, Xinyu Xing, and Gang Wang. CADE: Detecting and explaining concept drift samples for security applications. In *Proc. of the USENIX Security Symposium*, pages 2327–2344, 2021.

[139] Xinyang Zhang, Ningfei Wang, Hua Shen, Shouling Ji, Xiapu Luo, and Ting Wang. Interpretable deep learning under fire. In *Proc. of the USENIX Security Symposium*, 2020.

[140] Xuejun Zhao, Wencan Zhang, Xiaokui Xiao, and Brian Lim. Exploiting Explanations for Model Inversion Attacks. In *IEEE/CVF International Conference on Computer Vision*, pages 682–692, 2021.

[141] Deqing Zou, Yawei Zhu, Hai Jin, Hengkai Ye, Y Zhu, H Ye, Shouhuai Xu, and Zhen Li. Interpreting Deep Learning-based Vulnerability Detector Predictions Based on Heuristic Searching. *ACM Transactions on Software Engineering and Methodology*, 30, 08 2020.

A. Selected Literature

The literature related to *explainability* and *cybersecurity* has increased dramatically since 2014, see Figure 5. This literature is fragmented across various Computer Science domains. Table 4 provides a list of venues from where the reviewed literature was selected.

B. XAI Pipeline Design

We develop a modular XAI pipeline in Python (since it has in-built support for many popular models and explainers). The pipeline has three components: (1) The parser parses the input data (train and test) in either CSV or NumPy array format. The user can specify to the parser which feature fields should be read by means of providing a configuration file for the parser (2) The classifiers are implemented as a wrapper over the ML algorithms provided by `scikit-learn`. We currently support decision trees, logistic regression, explainable boosting machines, random forests, gradient boosting machines, and SVMs. The wrapper specifies the ML algorithm and its hyperparameters. (3) Similarly, the explainers are also implemented as wrapper functions and currently provide support for SHAP, LIME, LEMNA, and ELI5. The modules can be extended for added support of custom parsers, models and explainers. For the sake of reproducibility, the pipeline saves the model, predictions and explanations in a file.

TABLE 4. VENUES INVESTIGATED FOR LITERATURE DISCOVERY

Domain	Type	Venue
Cybersecurity	Conference	ACM Conference on Computer and Communications Security (CCS)
Cybersecurity	Conference	Asia Conference on Computer and Communications Security (AsiaCCS)
Cybersecurity	Conference	European Symposium on Research in Computer Security (ESORICS)
Cybersecurity	Conference	European Symposium on Security and Privacy (Euro S&P)
Cybersecurity	Conference	IEEE Symposium on Security and Privacy (S&P)
Cybersecurity	Conference	International Conference on Security and Privacy
Cybersecurity	Conference	Italian Conference on Cybersecurity (ITASEC)
Cybersecurity	Conference	Network and Distributed System Security (NDSS)
Cybersecurity	Conference	USENIX Security Symposium
Cybersecurity	Journal	Computers and Security
Cybersecurity	Journal	IEEE Transactions on Dependable and Secure Computing (TDSC)
Cybersecurity	Journal	IEEE Transactions on Information Forensics and Security (TIFS)
Cybersecurity	Journal	IEEE Transactions on Networks and Systems Management (TNSM)
Cybersecurity	Workshop	ACM workshop on Artificial Intelligence and Security (AISec) @ CCS
Cybersecurity	Workshop	ACM workshop on Wireless Security and Machine Learning
Cybersecurity	Workshop	AI-enabled Cybersecurity Analytics and Deployable Defense (AI4Cyber)
Cybersecurity	Workshop	IEEE Symposium on Visualization for Cybersecurity (VizSec) @ VIS
Cybersecurity	Workshop	Machine Learning for Cyber Security (MLCS) @ ECML/PKDD
Cybersecurity	Workshop	Workshop on Artificial Intelligence and Cybersecurity (AI-Cybersec)
Cybersecurity	Book	Malware Analysis using Artificial Intelligence and Deep learning (Springer)
Machine learn.	Conference	AAAI Conference on Artificial Intelligence
Machine learn.	Conference	ACM Conference on Knowledge Discovery & Data Mining (SIGKDD)
Machine learn.	Conference	Conference on Neural Information Processing Systems (NeurIPS)
Machine learn.	Conference	International Conference on Computer Vision
Machine learn.	Conference	International Conference on Intelligence Virtual Agents
Machine learn.	Conference	International Conference on Machine Learning (ICML)
Machine learn.	Conference	International Conference on Pattern Recognition
Machine learn.	Conference	International Joint Conference on Artificial Intelligence (IJCAI)
Machine learn.	Conference	International Joint Conference on Neural Networks (IJCNN)
Machine learn.	Journal	Advances in Intelligent Systems and Computing (Springer)
Machine learn.	Journal	Human-Intelligent Systems Integration (Springer)
Machine learn.	Journal	Nature Machine Learning
Machine learn.	Journal	Neural Computing and Applications (Springer)
Computer Sci.	Conference	ACM Symposium on High-Performance Parallel and Distributed Computing (HPDC)
Computer Sci.	Conference	Conference on Human Factors in Computing Systems (CHI)
Computer Sci.	Conference	International Conference on Enabling Technologies (WETICE)
Computer Sci.	Conference	International Conference on Human System Interactions (HSI)
Computer Sci.	Journal	ACM Computing Surveys
Computer Sci.	Journal	Annual Conference on Industrial Electronics Society (IECON)
Computer Sci.	Journal	Electronics (MDPI)
Computer Sci.	Journal	Expert Systems with Applications (Science Direct)
Computer Sci.	Journal	IEEE Access
Computer Sci.	Journal	Lancet Digital Health (Elsevier)
Computer Sci.	Journal	Procedia Computer Science (Science Direct)
Computer Sci.	Journal	Quality and Reliability Engineering (Wiley)
Software Engg.	Conference	ACM Joint European Software Engineering Conference and Symposium on Foundations of Software Engineering (ESEC/FSE)
Software Engg.	Conference	ACM/IEEE International Conference on Model Driven Engineering Languages and Systems (MODELS)
Software Engg.	Journal	ACM Transactions on Software Engineering
Software Engg.	Journal	ACM Transactions on Software Engineering and Methodology
Software Engg.	Workshop	International workshop on Continuous Software Evaluation and Certification (IWCSE) @ ARES

C. LEMNA Implementation

We based our implementation of LEMNA on the code by Warnecke *et al.* [129]. For the explanation generation, we use the following settings: $N = 500, K = 6, S = 10$. The values of N and K are based on the original LEMNA paper. We do not need fused Lasso since our features do not have a temporal structure. Therefore we set S to a high value, effectively turning off the fusing effect. We expect LEMNA to perform better on tabular data when using feature discretization. However, optimizing LEMNA is out of the scope of this work as we are only using existing methods for model debugging.

D. Explanations from the Tutorial

Figure 6 shows the global SHAP plot for the Gradient Boosting Machine (GBM) learned from the experimental dataset. Figures 7 and 8 show the post-hoc explanations for the false positive Netflow with and without the identified spurious features, respectively. Figures 9 and 10 show the post-hoc explanations for the false negative

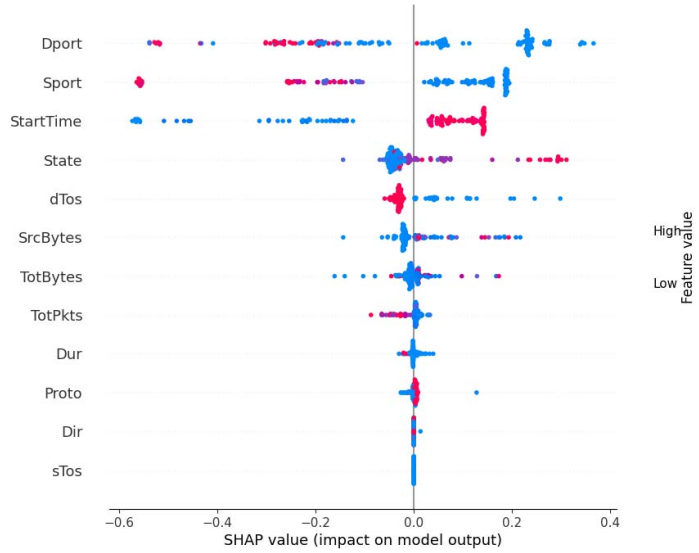

Figure 6. Global SHAP summary plot for the GBM. The y-axis shows the features ordered by their importance. For each feature, the Netflows are depicted as points, and their colour represents the contribution of the specific feature towards the model output.

Feature	SHAP Value
State = 54	0.2339
SrcBytes = 186	-0.1756
StartTime = 1313593252	-0.1210
Dport = 80	0.1121
Sport = 1703	-0.1113
dTos = 0	0.0465
TotPkts = 8	0.0408
TotBytes = 492	-0.0295
Proto = 0	0.0266
Dur = 8.96	-0.0247
Dir = 2	0.0
sTos = 0	0.0

Feature	Value	LIME Rule	Weight
Sport	1703	Sport=1703	0.17
StartTime	1313593252	1313537772.00 < Start...	0.14
Dport	80	Dport = 80	0.12
TotPkts	8	TotPkts > 4.00	0.05
Proto	0	Proto=0	0.03
TotBytes	492	271.50 < TotBytes <= 4...	0.02
State	54	State=54	0.01
Dir	2	Dir = 2	0.01
Dur	8.96	0.13 < Dur <= 9.01	0.01
SrcBytes	186	83.50 < SrcBytes <= 1...	0.0

Feature	Coefficient
dTos	1.0
sTos	2.46E-10
StartTime	4.53E-11
Sport	-4.33E-11
Dir	-4.06E-11
State	3.77E-11
Dur	2.07E-11
Proto	-1.93E-11
Dport	-5.96E-12

Figure 7. Post-hoc explanations for the False Positive Netflow (including spurious features). Left: SHAP, Middle: LIME, Right: LEMNA.

Feature	SHAP Value
State = 54	-0.3583
Dur = 8.96	-0.2450
TotBytes = 492	-0.1873
SrcBytes = 186	-0.1518
Dport = 80	0.0850
dTos = 0	0.0569
TotPkts = 8	0.0359
Proto = 0	0.0023
Dir = 2	0.0023
sTos = 0	0.0

Feature	Value	LIME Rule	Weight
TotPkts	8	TotPkts > 4.00	0.13
State	54	State=54	0.11
Dport	80	Dport = 80	0.10
Dur	8.96	0.13 < Dur <= 9.01	0.10
TotBytes	492	271.50 < TotBytes <= 4...	0.06
SrcBytes	186	83.50 < SrcBytes <= 1...	0.05
sTos	0	sTos <= 0.00	0.0
dTos	0	dTos <= 0.00	0.0
Proto	0	Proto=0	0.0
Dir	2	Dir = 2	0.0

Figure 8. Post-hoc explanations for the False Positive Netflow (after removing spurious features). Left: SHAP, Right: LIME.

Netflow with and without the identified spurious features, respectively.

Feature	SHAP Value
Dport = 3389	0.5387
State = 16	0.0495
Sport = 4505	-0.0488
StartTime = 1313571534	-0.0475
dTos = 0	0.0274
TotPkts = 10	0.0209
TotBytes = 1076	-0.0077
SrcBytes = 437	-0.0057
Dur = 60.95	0.0032
Proto = 0	0.0
Dir = 2	0.0
sTos = 0	0.0

Feature	Value	LIME Rule	Weight
Dport	3389	Dport = 3389	0.18
StartTime	1313571534	1313537772.00 < Start...	0.13
Sport	4505	Sport=4505	0.09
TotPkts	10	TotPkts > 4.00	0.07
State	16	State=16	0.04
Proto	0	Proto=0	0.03
SrcBytes	437	SrcBytes > 186.0	0.03
TotBytes	1076	TotBytes > 494.25	0.02
Dir	2	Dir = 2	0.01
Dur	60.95	Duration > 9.01	0.01

Feature	Coefficient
Dir	0.2574
Proto	0.2440
Dport	0.2404
Sport	0.2035
Dur	0.1680
State	0.1116
StartTime	0.0810
sTos	0.0241
dTos	-0.0179

Figure 9. Post-hoc explanations for the False Negative Netflow (including spurious features). Left: SHAP, Middle: LIME, Right: LEMNA.

Feature	SHAP Value
Dport = 3389	0.1542
SrcBytes = 437	0.0707
TotBytes = 1076	-0.0591
dTos = 0	0.0394
State = 16	0.0312
TotPkts = 10	0.0200
Dur = 60.95	-0.0153
Proto = 0	-0.0014
Dir = 2	0.0003
sTos = 0	0.0

Feature	Value	LIME Rule	Weight
Dport	3389	Dport=3389	0.20
TotBytes	1076	TotBytes > 494.25	0.15
TotPkts	10	TotPkts > 4.00	0.14
State	16	State=16	0.04
SrcBytes	437	SrcBytes > 186.0	0.03
Dir	2	Dir=2	0.01
sTos	0	sTos <= 0.0	0.0
dTos	0	dTos <= 0.0	0.0
Proto	0	Proto=0	0.0
Dur	60.95	Duration > 9.01	0.0

Figure 10. Post-hoc explanations for the False Negative Netflow (after removing spurious features). Left: SHAP, Right: LIME.