Generating End-to-End Adversarial Examples for Malware Classifiers Using Explainability

Ishai Rosenberg and Shai Meir and Jonathan Berrebi and Ilay Gordon and Guillaume Sicard and Eli (Omid) David
Deep Instinct Ltd
{ishair, shaim, jonathanb, ilayc, guillaumes, david}@deepinstinct.com

***Abstract*—In recent years, the topic of explainable machine learning (ML) has been extensively researched. Up until now, this research focused on regular ML users use-cases such as debugging a ML model. This paper takes a different posture and show that adversaries can leverage explainable ML to bypass multi-feature types malware classifiers. Previous adversarial attacks against such classifiers only add new features and not modify existing ones to avoid harming the modified malware executable's functionality. Current attacks use a single algorithm that both selects which features to modify and modifies them blindly, treating all features the same. In this paper, we present a different approach. We split the adversarial example generation task into two parts: First we find the importance of all features for a specific sample using explainability algorithms, and then we conduct a feature-specific modification, feature-by-feature. In order to apply our attack in black-box scenarios, we introduce the concept of transferability of explainability, that is, applying explainability algorithms to different classifiers using different features subsets and trained on different datasets still result in a similar subset of important features. We conclude that explainability algorithms can be leveraged by adversaries and thus the advocates of training more interpretable classifiers should consider the trade-off of higher vulnerability of those classifiers to adversarial attacks.**

I. Introduction

In recent years, the topic of explainable and interpretable machine learning (ML) has been extensively researched. Explainable machine learning can be used by the ML model users and developers, e.g., to debug prediction errors of an existing ML model on specific samples or to provide human explanations of models' decisions by highlighting the features that had the highest impact on a specific sample decision.

In this paper, we demonstrate the ability of an adversary to use the explainability of multi-feature types malware classifiers algorithms to produce the most important features for a known, white-box model. Then, due to the concept of transferable explanations, the same important features are relevant to the target black-box classifier. Therefore, by modifying existing malware's important features by the white-box model's ex-

single feature type: changing pixel's color in an input image, modifying words in an input sentence, etc. In those cases, modifying each feature has the same level of difficulty, because they all have the same feature type.

In contrast, in the cyber security domain there is a unique challenge which is not addressed by previous research: Malware classifiers (which gets an executable file as input and predict the labels of benign or malicious for each file) often use more than a single feature type (see Section I-A2). Thus, adversaries who want to subvert those systems should consider modifying more than a single feature type. Some feature types are easier to modify without harming the executable functionality than others (see Section I-A2). In addition, even the same feature type might be modified differently depending on the sample format. For instance, modifying a printable string inside a PE file might be more challenging than modifying a word within the content of an email, although the feature type is the same. This means that we should not only take into account the impact of a feature on the prediction, but also the difficulty of modifying this feature type [5]. In addition, some features are dependent on other features, meaning that modifying one feature requires modifying other features for the executable to continue functioning. For instance, adding strings to the file (as done in [4]) will necessarily impact the features dependent on printable characters, e.g., entropy.

The end result is that when adversaries want to modify a PE file without harming its functionality, the feature modification must be done manually. In this way, only features that are easy to modify, not dependent on other features who are challenging to modify (which is feature specific) would be modified. Thus, we would want to modify the smallest numbers of features, because each feature's modification requires a manual effort. Moreover, each modified feature can create a feature distribution anomaly that could be detected by anomaly detection algorithms (e.g., [6]). Therefore, the adversary aims to modify as little features as possible, even if he/she could modify