## Introduction

Cyber attackers are turning to document-based malware as suggested by many anti-virus (AV) vendors. Users are increasingly being warned more generally of the danger of executable files by browsers, email agents, or AV products, but documents such as PDFs are treated with much less caution and scrutiny because of the impression that they are static files and can do little harm.

However, over time, PDF specifications have changed. The added scripting capability makes it possible for documents to work in almost the same way as executable files, including their ability to connect to the Internet, run processes, and interact with other files/programs. The growth of content complexity gives attackers more weapons with which to launch powerful attacks and more flexibility to hide malicious payloads (e.g., encrypted files, files hidden in images, fonts, or Flash content) and evade detection.

A maldoc usually exploits one or more vulnerabilities in its interpreter to launch an attack. Unfortunately, given the increasing complexity of document readers and the wide library of component dependencies, attackers are presented with a large attack surface. New vulnerabilities continue to be found, with 137 published CVEs in 2015 and 227 in 2016 for Adobe Acrobat Reader (AAR) alone. The popularity of AAR and its large attack surface make it among the top targets for attackers. The collected malware samples show that many Adobe components have been exploited, including element parsers and decoders, font managers, and the JavaScript engine.

The continued exploitation of AAR along with the ubiquity of the PDF format makes maldoc detection a pressing problem, and many solutions have been proposed in recent years to detect documents bearing malicious payloads. These techniques can be classified into two broad categories: static and dynamic analysis.

Static analysis, or signature-based detection, parses the document and searches for indications of malicious content, such as shellcode or similarity to known malware samples. Dynamic analysis, or execution-based detection, runs the partial or entire document and traces malicious behaviors, such as vulnerable application programming interface (API) calls or return-oriented programming (ROP).

In the first half of this paper, we utilize machine learning (ML) techniques on document-specific attributes to identify embedded malware. Our approach addresses some of the shortcomings of existing techniques through the use of a broadly applicable mechanism to classify and characterize documents.

As part of our analysis, we show that while the use of documents as an exploitation vector can be an enabling mechanism for the attacker, it also provides additional detection opportunities. All of the data closely associated with malicious activities can be used to aid detection, regardless of whether the data utilized for detection are inherently malicious or not. The underlying premise and intuition of our study are that malicious documents do have similarities to other malicious documents; they also have dissimilarities to benign documents, regardless of the specific vulnerability exploited or the specific malware embedded in the document. We posit that features based on document structure and metadata are adequate for reliable document classification given that appropriate statistical methods are applied to these features. This ensemble classifier is also able to classify previously unseen variants.

Clearly, deployment of learning methods in any security-critical context requires that they can withstand potential attacks. The security of ML methods has been previously discussed from conceptual, methodical, and practical viewpoints. Typically, the security analysis of proposed learning-based techniques is carried out informally and is occasionally supported by experimental evaluation. From the practical perspective, the success of attacks against learning algorithms crucially depends on the amount of knowledge available to an attacker. Most of the previously reported successful attacks assume that the attacker has full knowledge of the learned model.

Still, it remains largely unclear what an attacker may learn about a learning-based method deployed “in the wild” and how this information can be exploited. To investigate this problem, we present the results of a case study we performed on a real learning-based model. For any submitted PDF file, the model provides a probabilistic estimate of its maliciousness. Our study addresses the case when an attacker attempts to evade detection by modifying the submitted PDF file so that its malicious functionality remains intact but the probabilistic score returned by the model is decreased.

To systematically explore the attacker’s options, we define an orthogonal set of evasion strategies reflecting various degrees of available knowledge. The general idea of our evasion technique is based on insertion of dummy content into PDF files that is ignored by PDF renderers but affects the computation of features. Once we can influence a subset of features, we develop algorithms for constructing attack instances. In the experiments, we evaluate the effectiveness of our strategies against our model.

In summary, this paper makes the following contributions:

* A new document dataset with 173,036 malicious files and 28,332 benign files.
* Identification of 133 useful and comprehensive static features for detection.
* A high accuracy rate of 99.82%, with a false positive rate of less than 0.01% for the learned model.
* Prediction time for single file maintains at a millisecond level.
* Development of an adversarial examples detection framework including adversarial example generation, robust model generation, and evasion detection.
* Presentation of a general model for practical assessment of the security of learning-based detection techniques; this model enables systematic exploration of various kinds of information leaks exploitable by an attacker and is applicable to systems that have a modifiable subset of features.
* We provide an open-source software framework for all experiments carried out in our study for independent verification and extension of our results.