MLsploit: A Framework for Interactive Experimentation with Adversarial Machine Learning Research

Extended Abstract

Nilaksh Das[1], Siwei Li[1], Chanil Jeon[1], Jinho Jung[1*], Shang-Tse Chen[1*], Carter Yagemann[1*], Evan Downing[1*], Haekyu Park[1], Evan Yang[2], Li Chen[2], Michael Kounavis[2], Ravi Sahita[2], David Durham[2], Scott Buck[2], Polo Chau[1], Taesoo Kim[1], Wenke Lee[1]

[1] Georgia Institute of Technology, Atlanta, GA, USA

{nilakshdas,robertsiweili,jinho.jung,schen351,yagemann,edowning3,haekyu,polo,taesoo}@gatech.edu

{chanil.jeon,wenke}@cc.gatech.edu

[2] Intel Corporation, Hillsboro, OR, USA

{chih-yuan.yang,li.chen,michael.e.kounavis,ravi.sahita,david.durham,scott.buck}@intel.com

ABSTRACT

We present **MLsploit**, the first user-friendly, cloud-based system that enables researchers and practitioners to rapidly evaluate and compare state-of-the-art adversarial attacks and defenses for machine learning (ML) models. As recent advances in adversarial ML have revealed that many ML techniques are highly vulnerable to adversarial attacks, MLsploit meets the urgent need for practical tools that facilitate interactive security testing of ML models. MLsploit is jointly developed by researchers at Georgia Tech and Intel, and is open-source (https://mlsploit.github.io). Designed for extensibility, MLsploit accelerates the study and development of secure ML systems for safety-critical applications. In this showcase demonstration, we highlight the versatility of MLsploit in performing fast-paced experimentation with adversarial ML research that spans a diverse set of modalities, such as bypassing Android and Linux malware, or attacking and defending deep learning models for image classification. We invite the audience to perform experiments interactively in real time by varying different parameters of the experiments or using their own samples, and finally compare and evaluate the effects of such changes on the performance of the machine learning models through an intuitive user interface, all without writing any code.

KEYWORDS

Adversarial ML, deep learning, interactive experimentation

OVERVIEW

MLsploit is a machine learning evaluation and fortification framework designed for education and research of adversarial ML. It is the first cloud-based tool that allows real-time interactive experimentation with state-of-the-art adversarial ML research through a web-based interface. By unveiling a host of vulnerabilities in the underlying techniques related to machine learning, recent research in the field of adversarial ML has cast a shadow on the deployment of ML models in safety-critical applications. Several works in this domain have also introduced robust defense techniques for these vulnerabilities. MLsploit aims to accelerate such research in this field by focusing on ML security related techniques and providing an interactive platform for researchers and practitioners to evaluate and strengthen the robustness of ML models in adversarial settings. We have also created a demo video for MLsploit, which can be viewed at https://youtu.be/hlzszoQVgD4.

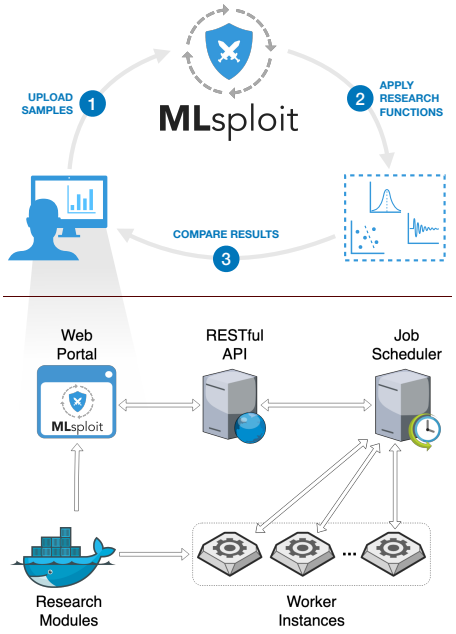

Figure 1: Top: Using MLsploit's interactive web interface, users can upload their own samples (e.g., APK files, images etc.), apply research functions to them (e.g., modifying APK files to bypass malware detection), and compare the results with previous experiments. Bottom: MLsploit system architecture. The frontend web portal gets data from a backend REST API. The experiments are run on separate instances by the job scheduler. The research modules are independently developed and deployed using Docker containerization.

The MLsploit system has a service-oriented architecture (SOA) that includes a web portal for users to interact with, and a RESTful API to further automate experiments. The research functions intetgrated in MLsploit can be thought of as modular components which can be combined in different arrangements to create the desired

* Authors contributed equally.

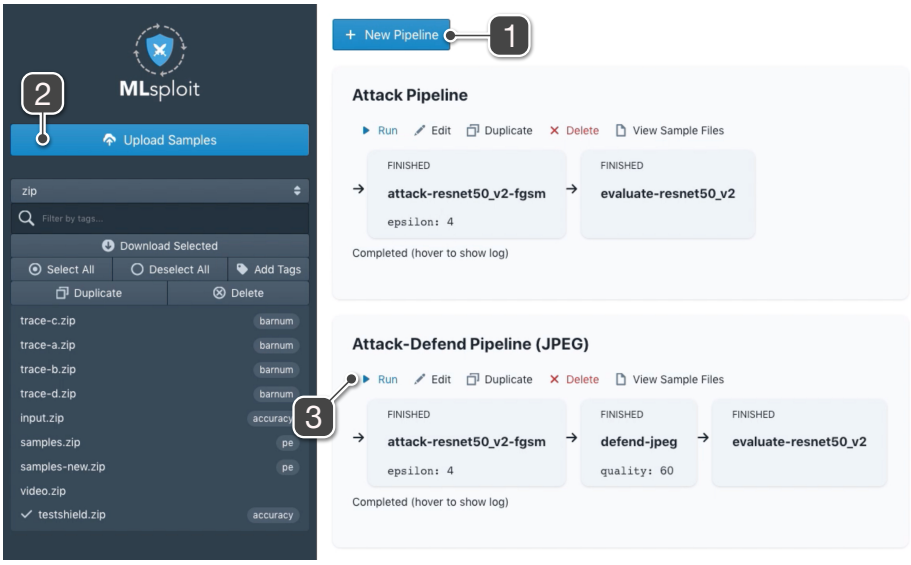

Figure 2: MLsploit web interface. (1) User creates pipelines by combining various research functions. Pipelines, with the selected parameters for each function, are shown in the right panel. (2) User uploads their own data samples to MLsploit, which are then listed in the left panel. (3) User runs an experiment by selecting from the uploaded files and pressing the "Run" button for a pipeline, which initiates a background job that passes the selected input through the various functions, and shows the output upon job completion. Output from previous runs are also shown for comparison.

experiment pipeline using an intuitive interface. Since MLsploit leverages Docker containerization in the backend, each component can be implemented in any language and on any platform, and MLsploit glues everything together through well-defined APIs. This flexible component design is agnostic to the underlying implementation of the machine learning function, and hence allows quick development for researchers as well.

MLsploit was developed at the Intel Science & Technology Center for Adversary-Resilient Security Analytics (ISTC-ARSA), which is a collaboration between Intel Corporation and Georgia Tech. The development of this tool was motivated by the growing need for quickly iterating over evaluation and comparison of the performance of machine learning models with conjunction to adversarial attacks and defenses. To further promote collaboration with other researchers, the entire MLsploit framework is open-sourced on GitHub (https://github.com/mlsploit).

Through this demonstration of MLsploit, we highlight the following contributions to the research community:

1. Interactive experimentation with adversarial ML research. MLsploit enables ML researchers and practitioners to perform experiments on their own data interactively, without writing any code, by combining plug-able components and services that implement various ML and security-related functions.

2. Enabling comparison of attacks and defenses across modalities. MLsploit currently integrates adversarial ML research modules that include bypassing the detection of Android, Linux and Windows malware, and attacks and defenses on image classification models. Users can compare the effect of varying different parameters for these techniques seamlessly through MLsploit.

3. Facilitating easy integration of novel research. The modularity of the MLsploit framework allows researchers to easily integrate their own research modules into the toolchain. We have also developed a python library that abstracts away the orchestration of the input files and experiment parameters from the web backend, and provides a simple API to the researcher for accessing these artifacts in their own python module. Hence, it becomes very easy to write and integrate new research functions for MLsploit, and keep the tool updated with state-of-the-art techniques.

ADVERSARIAL ML MODULES

MLsploit currently supports a host of adversarial ML modules spanning a diverse set of modalities, which we will demonstrate at this showcase and also invite the audience to experiment with:

- **AVPass**: leak and bypass Android malware detection [2].
- **Barnum**: defend models for document malware detection [3].
- **ELF**: bypass Linux malware detection with API perturbation.
- **PE**: create and attack models for Windows PE malware detection.
- **SHIELD**: attack and defend state-of-the-art image classifiers [1].

REFERENCES

[1] Nilaksh Das, Madhuri Shanbhogue, Shang-Tse Chen, Fred Hohman, Siwei Li, Li Chen, Michael E Kounavis, and Duen Horng Chau. 2018. SHIELD: Fast, Practical Defense and Vaccination for Deep Learning using JPEG Compression. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 196–204.

[2] Jinho Jung, Chanil Jeon, Max Wolotsky, Insu Yun, and Taesoo Kim. 2017. AVPASS: Leaking and Bypassing Antivirus Detection Model Automatically. *Black Hat USA Briefings (Black Hat USA), Las Vegas, NV* (2017).

[3] Carter Yagemann, Salmin Sultana, Li Chen, and Wenke Lee. 2019. Barnum: Detecting Document Malware via Control Flow Anomalies in Hardware Traces. In *Proceedings of the 22nd Information Security Conference.*