A Artifact Appendix

A.1 Abstract

We developed a Web platform and an API client that allow users to retrieve the Expected Exploitability (`EE`) scores predicted by our system. The system gets updated daily with the newest scores.

We implemented an API client in python, distributed via Jupyter notebooks in a Docker container, which allows users to interact with the API and download the `EE` scores to reproduce the main result from the paper, in **Figure 5(a)** and **Figure 8(a)**, or explore the performance of the latest model and compare it to the performance of the models from the paper.

A.2 Artifact check-list (meta-information)

- **Program:** Docker (tested on v20.10.8)
- **Run-time environment:** UNIX-like system
- **Metrics:** Rrecision, Recall, Precision-Recall AUC, TPR, FPR, AUC.
- **Output:** Plots from the paper, reproducing and expanding Figure 5(a) Figure 8(a).
- **Experiments:** Running Jupyter notebooks.
- **How much disk space required (approximately)?:** 4GB
- **How much time is needed to prepare workflow (approximately)?:** 15 min
- **How much time is needed to complete experiments (approximately)?:** 30 min
- **Publicly available?:** The website and client code are publicly available. The API requires an API token which we provide upon request.

A.3 Description

A.3.1 Web Platform

The Web platform exposes the scores of the most recent model, and offers two tools for practitioners to integrate `EE` in vulnerability or risk management workflows.

The *Vulnerability Explorer* tool allows users to search and investigate basic characteristics of any vulnerability on our platform, the historical scores for that vulnerability as well as a sample of the artifacts used in computing its `EE`. One use-case for this tool is the investigation of critical vulnerabilities, as discussed in Section 7.3 - `EE` for critical vulnerabilities.

The *Score Comparison* tool allows users to compare the scores across subsets of vulnerabilities of interest. Vulnerabilities can be filtered based on the publication date, type, targeted product or affected vendor. The results are displayed in a tabular form, where users can rank vulnerabilities according to various criteria of interest (e.g., the latest or maximum `EE` score, the score percentile among selected vulnerabilities, whether an exploit was observed etc.). One use-case for the tool is the discovery of critical vulnerabilities that need to be prioritized soon or for which exploitation is imminent, as discussed in Section 7.3 - `EE` for emergency response.

A.3.2 API Client

The API allows clients to download historical scores for a given vulnerability (using the `/scores/cveid` endpoint), or all the prediction scores for a particular model on a particular date (using the `/scores/daily` endpoint). The API documentation describes the endpoints and the parameters required for each call, and provides example code for clients to interact with the API.

A.4 How to access

The Web platform is available at https://exploitability.app/. The API and the client code are available at https://api.exploitability.app/.

A.5 Installation

The practitioner tools are available on the Web platform. To use the API, clone the code repository, point a terminal to that folder and run `bash docker/run.sh`. This will create the Docker container and spawn a Jupyter server. Use the URL displayed in the console to open a browser session within that container.

A.6 Evaluation and expected results

To reproduce the results from the paper, open and run the following notebook: `reproducibility_plot_performance.ipynb` In an API key is provided, this will download the required scores used in the paper, cache them in various files in `scores_reproducibility_download/`, and use these files to compute the performance of `EE` and baselines. The output consists of 2 figures, which correspond to Figure 8(a) and Figure 5(a) in our paper.

To evaluate the latest model, open and run the following notebook: `latest_plot_performance.ipynb` In an API key is provided, the notebook will download all the scores produced by our latest model on `2021-10-10`, cache them into a file in `scores_latest_download/`, and use this file to compute the performance of `EE`. The output consists of 2 figures which are comparable to Figure 8(a) and Figure 5(a) in our paper.

As of December 2021 we observe that the performance of our latest model predicting `EE`, computed before `2021-10-10`, is very close to the performance reported in the paper (0.69 PR AUC / 0.96 AUC for latest model vs 0.73 PR AUC / 0.84 AUC in the paper), demonstrating that our online predictor is functional and in line with the claims from the paper. The performance of the latest model on other dates can be computed by changing the `SCORES_DATE` variable and re-running the notebook.

A.7 Experiment customization

When running `latest_plot_performance.ipynb`, users can customize the `SCORES_DATE` variable in the notebook to observe the performance of our model on different dates.