Machine Learning Engineer Nanodegree

Capstone Proposal

Kenneth R. Farr III

July 28th, 2018

Predicting Cyber Security Exploits from Vulnerabilities

Domain Background

Cyber Security is the domain that focuses on cost-effectively reducing losses in and around digital assets. This includes accidental losses caused by mistakenly emailing confidential information, and malicious losses such as a stolen mobile device or a breached database. A popular modeling technique for computing the risk (dollars lost per year) of an asset is the Factor Analysis of Information Risk (FAIR) ontology. This model accepts as input the frequency of a loss event (how often a loss is expected to occur) and the loss magnitude (when a loss occurs, how much does it cost the organization).

A Cyber security industry standard exists to catalog all known vulnerabilities: Common Vulnerabilities and Exposures (CVE). The CVE list details every known vulnerability and applies attributes to them, such as level of involvement to exploit, closeness to the asset (physical presence, adjacent network, internet) etc. These elemental attributes are used to derive a Common Vulnerability Scoring System (CVSS) score.

The anual Verizon Data Breach Investigations Report (VDBIR) is highly regarded by Cyber Security Professionals and in the 2015 report, page 21 stated that "...we agree with RISK I/O's finding that a CVE being added to Metasploit is probably the single most reliable predictor of exploitation in the wild". Metasploit DB (MDB) contains publically available exploits and is freely searchable.
As a member of a Research and Product Development team for a leading Cyber Security Risk Quantification Company, this kind of research (predicting exploits) is both interesting and important to our work.

As a member of a Research and Product Development team for a leading Cyber Security Risk Quantification Company, this kind of research (predicting exploits) is both interesting and important to our work.

Problem Statement

In a business landscape, prioritizing capital investments is critical to maintaining a competitive advantage. A portion of all capital expenditures include risk reduction, and cyber security is no exception. It would be cost-prohibitive for a

business to attempt to effectively patch every system against every vulnerability, therefore a business must prioritize which vulnerabilities get patched.
With over 5000 new CVE entries per year, a business must determine which vulnerabilities affect them, and by predicting which of these 5000 will most likely result in a usable exploit the business can prioritize their patching efforts.

Datasets and Inputs

As with any prediction algorithm, honest inputs and a ground truth are required for training. CVE entries will be used as feature inputs with a boolean output to indicate whether or not the CVE ever made it into the Metasploit DB.
Each CVE entry consists of defining attributes that derive CVSS scores. The goal is to utilize the CVSS ordinal classification attributes as inputs: Attack Vector, Attack Complexity, Privileges Required, User Interaction, Scope, Confidentiality, Integrity, Availability.
The entire list of CVEs can be downloaded in an XML format by year, this data will be extracted and placed in a frame along with the CVSS attributes. Each CVE will be searched in Metasploit (by downloading their database) and an attribute added to indicate whether or not the CVE has an active entry in Metasploit.
Some research will need to be performed to gauge the length of time for a CVE to be added to the CVE list and an entry in the Metasploit DB in order to avoid training on CVEs that have not been given adaquate time to be added to the Metasploit DB.

Solution Statement

To aid in vulnerability remediation, this research will attempt to design a machine learning algorithm that will take as input a CVE element, including CVSS attributes, and predict whether or not this CVE will eventually be added to the Metasplit DB.

Benchmark Model

The percentage of CVEs that make it into the Metasploit DB can be used as a rudimentary baseline model. If perhaps 2% of all CVEs make it into Metasploit, then each CVE predicted will have a 2% chance of being classified as potentially being added into Metasploit. Precision and Recall will be utilized to gauge the effectiveness of the final model.

Evaluation Metrics

While reducing both Type 1 and Type 2 errors is important, it is better that the model predicts more False Positives than False Negatives. As such, the model

will be designed to allow for a lower precision with a higher recall.
The data will be broken into train, cross-validate, and test data sets by selecting randomly. While perhaps naive, the model will not assume that any earlier CVEs will affect any later CVEs.

Project Design

1) Read in each years worth of CVEs in XML using Python. Create a Pandas dataframe for each CVE with the CVE ID and each CVSS attribute as a feature. Save this dataframe as processed data.
2) Download the Metasploit package and research how to extract the required data from it. This package is a github repository that will allow you to search for exploits by CVE ID. This interface has been tested online via their website, however extracting the data from the downloaded github repository has not been proven yet, though there's high confidence that it will be successful.
3) Create two new columns per CVE in the Pandas dataframe, one to indicate if there is ANY entries in the Metasploit database, and one to indicate the date it was added.
4) For all CVEs with a Metasploit entry, calculate the average time it took the CVE to enter into Metasploit once the CVE was added. This time will be used to trim all newer CVEs that should not be expected to be in the Metasploit database yet.
5) Separate the CVEs into training, cross validation, and testing sets. Ensure each set contains a proportional number of positive and negative cases.
6) Start with a simple logistic regression for a baseline, attempt hyperparameter tuning.
7) Experiment with SVMs in the same method
8) Experiment with ensemble decision trees
9) Write up the results