**Machine Learning Engineer Nanodegree**

**Capstone Proposal**

Thiago Nascimento

January 6, 2019

## Proposal

### Domain Background

### Use of Machine Learning Methods for Intrusion Detection Systems

The internet is now something invisible and inherent to our civilization, like electricity. Almost every human being tends to spend part of his day on communications app, device or tool. We constantly applying our social skills thru the virtual realms. Computer networks are the backbone of all digital communications being made around the world today. Smartphones, notebooks, smart-homes and any connected device exchanges data thru one of these networks. Although they may use 'different' protocols, they are all subject of the same networks and routers. The internet traffic is huge. IPV6 is already working and supporting this massive information demand. There is something between 22 and 26 billion devices connected to the internet [1]. This numbers tend to grow massively as internet of things becomes the rule. More traffic means more treats, sophisticated viruses and malwares, zero-day attacks, and even, the now popular term, “cyber war”.

It is not new to use machine learning techniques within offensive and defensive security areas. In this project, various classifiers are used to identify unknown and / or unwanted traffic. Their results are compared with each other and with the more common approaches, the signature-based systems. We choose a well-known dataset in the research of Intrusion Detection techniques, the NSL-KDD. An intrusion detection system is used to monitor the traffic in real time, classifying the treats per level. It can also become an autonomous system that drops these anomalous, inappropriate and / or unauthorized packets, an IPS (Intrusion Prevention System). An IPS can be seen as an extension of and IDS.

Being a computer networks engineer and working on a company that develop devices for network defensive security based on dynamic automated over-the-air update signatures and rules, personal and professional motivations are just relevant and clear enough.

## Problem Statement

On enterprise networks connected to internet (default model for any-size company), identifying incoming data is an old problem. When it comes to sensitive information and always-on systems, things get a little more complicated, as this scenario is highly dependent of specialists and real-time human intervention. Most of the main approaches present another very common downside, higher levels of inaccuracy.

## Datasets and Inputs

The NSL-KDD [2] is a prime tool for research and improvement of intrusion detection approaches. The good quality of this data presents an ideal scenario for offline works and benchmarks. The NSL-KDD dataset is an improved version of the KDD'99 dataset [3].

The inherent drawbacks of the KDD-cup'99 dataset has been, almost all, handled in the NSL-KDD. To address some of the corrections refinements we can quote:

- Redundant and / or duplicated records was removed from the train and the test set, which enable the classifiers with better detection rates to perform un-biased results on more frequent records.
- There are enough records in the train and test sets, which makes it affordable to run the experiments on the complete set without the need to randomly select a small portion or use some technique like random cross validation. This enables evaluation results of different works to be consistent and comparable.
- The number of selected records from each difficulty level group is inversely proportional to the percentage of records in the original KDD data set. As a result, the classification rates of distinct machine learning methods vary in a wider range, which makes it more efficient to have an accurate evaluation of different learning techniques.

The amount of details is huge and very well documented, but we will not discuss it just now (attributes details, attributes names and description of each one). On exploring the dataset, we can count forty-one attributes that we will treat as features of the flow. Each one has an assigned label as a type of attack or normal traffic. We can access, besides what was considered above, detailed information of all 41 attributes, classes of the network connection (1 normal class and 4 attack class) and description of each attack class.

The NSL-KDD is not perfect. It still has some problems as it does not match exactly a real network. However, it is the maybe the most effective benchmark for the purpose of researching and comparing different methods and techniques for flow detection.

## Solution Statement

Researches explore lots of different approaches for the problem of traffic identification. Some known strategies are discussed below.

Port-based (working with predefined ports). That method is maybe the older and is almost totally deprecated nowadays as it is simple to not follow any rule of port registration.

Signature-based is a relatively newer method (from 2002 until now). That method works with a part of the payload of the packets, which is normally static and unique for application layer. That's is very similar to strategies adopted by antivirus systems and it presents a relatively low error (bellow 10%). But as the specifications change and new protocols arrive, it demands a lot of time from specialists on identifying and validating signatures.

Solutions based on automated classification statistical methods are getting popular, as these approaches depend massively on the features learned from the traffic flow and machine learning strategies. On this project, the features are fed into different classifiers, such as Ada-Boost (with decision trees), Random Forest, Naive Bayes, SVM, and Neural Networks. To some of them, we will also apply the Random Projection technique. Also, we may apply some clustering technique for completely unknown data. For application, we expect a real-time model that can work on general purpose networks. However, the training step is offline and time consuming. And the key point is the selection of the features.

## Benchmark Model

The NSL-KDD dataset is used to estimate the performance of intrusion detection systems. The good quality of this data presents an ideal scenario for offline works and benchmarks. For this project, will be considered the values related to default approaches and the results achieved for various machine learning techniques, included comparisons between them.

For most common techniques, are being considered port/address-based and signature-based. For machine learning, at first, are being considered classification, association, hybrid and clustering methods (if totally unknown traffic).

All results can easily be compared to default methods as they are all published for this purpose.

## Evaluation Metrics

At this point, the metrics that will be used are defined below. Most of them, based on the table of a confusion matrix [4].
During the project is probable that more metrics will be added, as the different metrics can help on visualizing the results.
Percentage of successful prediction (PSP - The ratio of successful instances classified divided by the total numbers of actual instance).

- True Negatives (TN - Total numbers of normal packets correctly classified).
- True Positives (TP - Total numbers of malicious packets correctly classified).
- False Negatives (FN - Total numbers of malicious packets incorrectly classified as normal packets).
- False Positives (FP - Total numbers of normal packets incorrectly classified as malicious packets).
- Detection rate (DR - The ratio of total numbers of attacks detected divided by FP plus TN).
- Accuracy (ACC - The ratio of TP plus TN divided by of FP plus FN).
- Precision rate (PR - The ratio of TP divided by TP plus FP).

## Project Design

### *Data cleaning / Dataset Selection*

Between the Intrusion Detection Evaluation Dataset (CICIDS2017), KDD'99 from kddcup99 and NSL-KDD, besides some other datasets used for network traffic purposes, we choose NSL-KDD for reasons already described on the 'INPUT' part of this work.
For data cleaning, we searched for duplicated data, irrelevant data, structural errors, typos, inconsistent capitalization, mislabeled classes or redundant classifications, unwanted outliers and missing data (categorical and numerical). For some of these steps we can simply drop the data, for others we better correcting the data.
Although this is a good practice, most of the steps taken were taken as a precaution, in view of an extremely organized dataset as explained on the introduction and 'input' and 'domain' parts.

## *Pre-processing*

Discretization is essential on obtaining better results as it converts continuous features to intervals and then to discrete values. Normalization is done as a requirement because certain classifiers produce a better accuracy rate on normalized dataset.

## *Feature leaning / feature selection / dimensionality reduction*

Although extremely useful, the feature learning step is not necessary, as we think that will be more productive to reduce the number of features / dimensions.
Correlation based Feature Selection method is used in this work to reduce the dimensionality of the features available. With the same goal, we test different techniques, PCA - Principal Component Analysis and Random Projection.

## *Training data / testing data*

The dataset is already organized in train and test subsets, so there is no need to make any adjustments for this purpose. Maybe an evaluation subset can be useful for step by step analysis.

## *Algorithms and methods*

For this part, we will use pure, hybrid and associated classifiers. For now, we are considering Naive Bayes, SVM (support vector machine), Decision Trees and ANN (artificial neural networks). Some ensemble methods as AdaBoost and Random Forest. All them with and without previous dimension reduction steps.

## *Experimental setups and results*

Comparison between setups with different algorithms and feature engineering methods.

## *Conclusion and future works*

Relevant observations and suggested future changes on setups and new works.

[1] https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide
[2] https://www.unb.ca/cic/datasets/nsl.html
[3] http://archive.ics.uci.edu/ml/datasets/kdd+cup+1999+data
[4] https://www.dataschool.io/simple-guide-to-confusion-matrix-terminology/