vehere

WHITEPAPER

Machine Learning Implementation

Authors: Prasenjit Mandal & Puja Sarkar
prasenjit.mandal@vehere.com & puja.sarkar@vehere.com

Modern-day Security and Risk professionals, especially ones managing Security Operations Centre, are inundated with tasks pertaining to compliance, threat hunting, risk management and business/application security. With a scarcity of skill and time, it is often felt there should be more assistance to SOC professionals, enabling them to automate some of their tasks. This is especially true when it comes to being proactive in the overall approach – i.e., detect (potential) issues and work towards their resolution.

One such technology that has taken centre stage is the application of Artificial Intelligence/Machine Learning to detect issues where none may seemingly exist. Artificial Intelligence or Machine Learning detects existing or latent risks by analyzing volumes of data that is humanly impossible given their amount.

Vehere PacketWorker implements an unsupervised Machine Learning model that is based on Topic Modelling.

Introduction

The following few sections introduce a relatively novel concept – Topic Modelling and how Network data gets transformed to be used with Topic Modelling algorithm. The latter part of the document provides technical insight for the more serious reader.

Foundation

Topic Modelling

In Machine Learning, Topic-Modelling is a form of text mining, employing unsupervised statistical techniques to identify patterns in a corpus or large amount of text. Anything that does not fit the pattern, is an anomaly.

A general perception is that a document will touch upon a few topics. Therefore, one would expect some words to appear in the document more frequently than other words. This basis allows Topic Modelling algorithms to group words in the document to different topics. Expanding the same logic to a group of documents, the algorithm can group words from different documents into topic buckets to determine similarities. Words that do not fit into the topics are considered outliers.

Why use Unsupervised Machine Learning algorithm?

Prime drivers for using Unsupervised Learning:

- Ability to discover almost all kinds of unknown patterns in data.
- Find attributes that are useful for categorization.
- Real-time.
- Easier to implement although tuning takes time.

PacketWorker employs the use of unsupervised algorithms to analyze Network Traffic. Special attention is given to DNS Traffic since it contains names that can reveal curious insights into user behavior.

Three points are important going forward,

- Documents
- Topics
- Words

Let us try to understand how does this concept map to Network Traffic.

How does Topic Modelling work on Network Traffic?

PacketWorker views recorded "information" pertaining to each IP Address as a document. So, sessions pertaining to an IP Address are documents. There may be "n" documents in a network of "n" hosts at the time of processing. The system performs multiple processing cycles each day to periodically flag anomalies.

So, to summarize,

Document = Sessions of a given IP Address. For example, Source IP Address.

How does document get its words?

If a document is sessions (note, it is plural) of an IP Address, and a document is a collection of words, a word is a singular session entry. It is that simple. Hence,
Word = A single Session.

Multiple sessions for the same Node (IP Address) at the time of processing form a document with "w" number of words pertaining to the document "D",

D1 = Sessions of Node 1
D2 = Sessions of Node 2
D3 = Sessions of Node 3
Dn = Sessions of Node n

Each document is made up of words which are nothing but session entries. Think of the following scenario –

D1 = {W1, W2, W3, W4, W5, ..., Wn}
D2 = {Wa1, Wa2, W3, W5, ..., Wan}
D3 = {w1, w2, w3, ... Wn}
Dn = {W1, w1, W3, w4, Wa1, ..., Wx}

What is Topic?

This is the interesting part. For processing of network traffic using Topic Modelling, Topic is Behavior. We assume that a finite number of topics "t" can help classify almost all behavior exhibited in a network. In the case of PacketWorker, the default number of topics is 20.

The purpose of the analysis now is to classify documents D1 to Dn consisting of different words into these 20 topics. Anything that does not fit the pattern into these 20 topics is an anomaly.

The fundamental paradigm behind this rationale is that users in a group have similar behavior (application usage, timings, etc.) and machines of network will exhibit similar behavior (accessing resources – Domain controllers, file servers, printers, etc.). A session that does not pertain to a pattern will be flagged as anomalous

How are sessions converted into Words?

Conversion of network sessions into words is the process of transforming sessions into a structure wherein the word

(representing a session) must preserve enough information to deliver anomalies during processing. The anomaly may imply - malicious behavior, change of temporal usage patter or, malfunction among other inferences.

Session conversion to word is done with an intent to ensure that similarities overlap across documents (IP addresses) so that processing the resulting text corpora produces meaningful results. Consequentially, transforming a session into word involves:

Identifying Features

Two sessions from the same client to the same application server will still have differences – packet lengths may change, underlying MAC Addresses of the intermediate hops may not be same, payload size and, session averages might vary owing to network/usage conditions. The challenge is to ensure that similarities are maintained consistently and represented in a manner that grouping of words to topic during the processing yields best results. Some examples of features that make up the word representing a session are –

- Protocol Information – This is generally the protocol number (6 for TCP, 17 for UDP, 1 for ICMP).
- Port Information – This is generally the service port identified by the direction of the first packet of the session (80 for HTTP, 443 for HTTPS).
- Initial Flag – This is the first flag of the session (S for SYN, SA for SYN-ACK, Z for no Flag).
- Transmitted Bytes – Since this number can vary, it is binned based on a "log-base-x" logic to produce numbers (0, 1, 2, 4, etc.).

Merging features to form word

The features identified and normalized are concatenated to form words. So, the word for an SSL session may be 6-443-S-3.

Let us summarize before moving on

The table below summarizes the process of transformation of network traffic to a set of documents and words. For the processing, the anomaly detection logic assumes a finite number of topics, default 20.

What gets used?	As what?
Source IP Addresses	Documents
Sessions	Words
Topic	Behavior
L2 – L7 values of the IP Header & Payload aggregated for a session	Features that are joined to form a word

How is DNS different?

DNS sessions contain the query name, status, response code and, the intermediate lookups (canonical names, response types, etc.) before a response is received. To detect latent inferences from among such traffic consisting of strings largely, these strings need to be normalized. PacketWorker employs novel techniques where construct of user queries is normalized to deliver a uniform set of features for words that can then be submitted to topic modelling. An example set of features for DNS Session are –

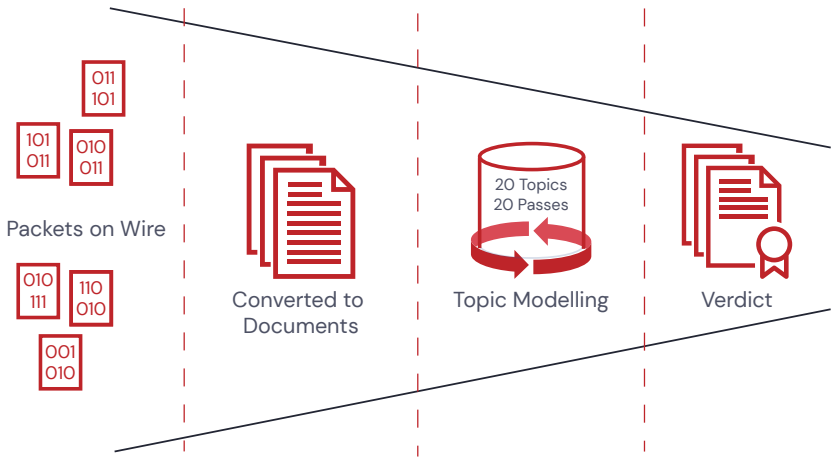
- Length of domain suffix binned using "log base-x" formula (0, 1, 2, 4, ...)
- Entropy of the query binned using "log base-x" formula (0, 1, 2, 3, ...)
- Non-empty transmitted bytes binned using "log base-x" formula (0, 1, 2, ...)
- Non-empty received bytes binned using "log base-x" formula (0, 1, 2, ...)
- Response code (0 = success, any other integer = error)

So, for an access to www(.)example(.)com, the word structure might look like 1-1-2-3-0.

Processing for Anomalies

PacketWorker employs a sliding-window method to keep a record of past 48-hours of activity at each processing interval. New dataset representing user activity between the previous processing interval and the current one is merged with the historical dataset, records older than 48-hours are expunged and, the resulting dataset (documents and words) submitted to the topic modelling process to infer latent insights; essentially, anomalies. To ensure that the results are relevant, the topic modelling process performs 20 sequential passes over the data set before delivering an outcome. The outcome is a set of scores between 0 and 1 with 0 implying lower outlier and 1 implying higher outlier. For the sake of simplicity, this data is further compartmentalized using equidistant binning method to deliver a final verdict between 0 to 100.

Sessions with scores less than 5 and, higher than 95 should be considered anomalies (although the user is free to tune the score for a better outcome) and investigated.



Field Input

Machine Learning is good at identifying hidden behavior that skips even the most trained human eye. However, it is the application of context that qualifies the anomaly as an event worth investigating. What is normal for one environment may be a serious offence for another. Therefore, it is important for Machine-Learning process to intake human input about the behavior to eliminate noise.

PacketWorker's implementation of Topic Modelling allows a site-wide white-list to set-up which uses the input to deprioritize notifications based on time-of-day, node's IP Address, DPI-Application/suffix.

Key Features of PacketWorker Machine-Learning Implementation

- Node and Network baselining over a 48-hour period.
- Pseudo-streaming mode of processing to detect anomalies in near-real time.
- Auto-profiling of network and node behaviors – comparing past with current.

Benefits

- No data/metadata or, word is shared with any external entity including Vehere's cloud. 100% localized analysis.
- Portable to run from a dedicated instance ingesting larger network data not just from the point of capture but including data from external sources (May require additional purchase. Please check with your Vehere representative).

Value

This model can identify most suspicious or unlikely network events in the observed network. If any types of attack or unusual activity happen, then that will be identified enabling greater visibility into unlikely events that happen across a wide timespan and usually escape human attention.

Technical Details

Machine Learning for DNS Analytics

Implementation of Machine-Learning for DNS Analytics in PacketWorker is a form of quasi-supervised anomaly detection that uses topic modelling to infer common query behaviors and build a model of DNS query behaviors for each node. Each DNS query is assigned an estimated probability (henceforth, the query's "score"). The queries with lowest & highest scores are flagged as "suspicious" for further analysis. PacketWorker employs Latent Dirichlet Allocation (LDA) model. The probability distributions that arise from an LDA model, generates an anomaly score that can be assigned to words of a document. In the case of DNS Analytics, "Words" are queries generated at a given "time" and, documents pertain to the Source IP Address. Consequentially, the model analyses a text corpus that has "N" documents (where there is one document for every node on the network) and each document has "W" words with each word implying time and query.

The outcome of this model is anomalous word "w" for a document "d". In simple terms, this is the DNS query originated by a node on the network which the model identifies as anomalous given the historical perspective of this host and the network in general.

Machine Learning for Network Behavior Anomaly Detection

Implementation uses topic modelling to infer common network behaviors and build a behavioral model for the network and, for each node present on the network. Just as with Machine Learning for DNS Analytics, latent Dirichlet Allocation (LDA) model is used to generate an anomaly score that can be assigned to words of a document.

In the case of Session Analytics, "Words" are sessions between any two given nodes at a given "time" and, documents pertain to both – Source Node and, Serving Node (Source and Destination IP Addresses). Consequentially, the model analyses a text corpus that has "N" documents (where there is one document for every communicating node) and each document has "W" words with each word implying time and session record for source and for destination.

The outcome of this model is anomalous word "w" for a document "d" for a "Pivot Node". In simple terms, it is the session between a "Pivot Node" and another host which the LDA model identifies as anomalous given the historical perspective of the "Pivot Node's" communication and the "Application's" general behavior on the network.

A "Pivot Node" is the host that is perceived to be the trusted host for the network. Detection of Pivot Node happens based on the system configuration or by the general distribution of network traffic across IP Networks.

The "words" for Session Anomalies are generated for each session and are different for the Source IP Address and, the Destination IP Address.

Glossary

Analysis Interval: Every 15 minutes.

Document: Communicating Endpoint's "Source IP Address" & "Destination IP Address" form "Source Document" and, "Destination Document" respectively.

Word: For each session, a set of features are concatenated together to create a word and each session is represented by its generated words. A "Source Word" for assignment to "Source Document" and, a "Destination Word" for assignment to "Destination Document" is generated.

Topic Modelling: "Source Documents" and "Destination Documents" consisting of their words form the input to the process. The classification process attempts to compartmentalize host and network behavior among a set of 20 topics and executes 20 sequential passes over the dataset.

Topic Modelling Outcome: A pair of numbers (between 0 and 1) implying probability of a particular word belonging to a specific topic - "Source Score" and "Destination Score".

This outcome is normalized into a set of 100 buckets (score between 0 to 100) using "equidistant binning" method to generate a distance score.

Normalization Formula:

bin_distance = ((max([score]) – min([score])) + 0.00000000001) / 100

distance_score = rounddown ((score – min ([score]))) / $bin_distance)

Inference of the distance score: It is the normalization of scores, to a scale between 0 and 100. This score signifies that the respective "word" representing a session is an outlier for the "document"; implying that this session behavior is an anomaly for the given host and hence is flagged when "source score" and "destination score" both cross the threshold boundaries.

Alerting: Alerts are raised for distance scores less than 5 or, greater than 95. This is tunable.