WHITE PAPER

exabeam

APPLYING DATA SCIENCE TO USER AND ENTITY BEHAVIOR ANALYTICS

Derek Lin,
Chief Data Scientist, Exabeam

This paper discusses the application of data science within a User and Entity Behavior Analytics (UEBA) product to address cyber threats. To provide concrete examples, we refer to the Exabeam Security Intelligence Platform, which implements UEBA functionality. In concept, a UEBA system such as Exabeam's monitors network entities' behaviors within an enterprise and flags unexpected behaviors worthy of investigation. While the benefits are understandable, there are many challenges. In this paper, we'll focus on the data analytics capabilities that have proven to work well in the field for a large number of customers with different environments. In the following sections, we first introduce the guiding philosophy in building Exabeam's data analytics and then describe the core statistical analysis employed. We then discuss how machine learning is used for context estimation, detection, and false-positive control.

Background

The Session Object

At Exabeam, data analytics begins with Stateful User Tracking™, which allows us to organize user events into sessions. The Exabeam system stitches raw event logs and other data, such as endpoint data, DLP data, badge readers, etc, into a unique session object. Exabeam builds session objects for every user on the network and every session (logon through logout) for each user. A session defines where a logical collection of events starts and ends. Various statistics in the system are organized and based on the concept of sessions. As such, sessions are the core informational units for learning and scoring, upon which Exabeam data analytics are built.

Combining Statistics, Machine Learning, and Security Research

A well-tuned statistical analysis system is at the heart of the Exabeam UEBA product. Our security researchers define a collection of more than a hundred statistical indicators for users, assets, peer groups, applications, network locations, etc. An anomaly is triggered based on a statistical model and is given an expert-assigned risk score, which encodes critically-important security knowledge. Without the encoded knowledge, any pure anomaly-based detection system based on unsupervised learning will suffer from a high false positive rate, rendering it impractical for field deployment. Combining expert knowledge and data analytics is particularly advantageous because it is intuitive and easy to use for analysts of all levels. Neither a purely expert-driven system nor a purely data-driven approach, this hybrid method has proven to work well in production (figure 1).

Simplicity is King

Many of the attempts to create a machine-learning system for behavioral analytics have exposed too much complexity to the end user. In our experience, a good UEBA system – like an iPhone – hides the complexity from the user wherever feasible. Balancing power and simplicity is the main challenge for machine learning-based UEBA system. Exposing machine learning output directly to end users without useful explanation only forces analysts to do more work to investigate; it doesn't reduce the workload. Despite its underlying complexity and sophistication, a good machine learning-based UEBA must strive to be simple to use, easy to configure, and with easily-explained output. Simplicity to users is king in the world of UEBA. This philosophy guides Exabeam's analytics design and choices.

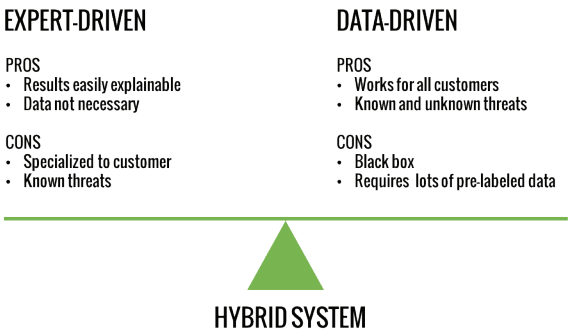


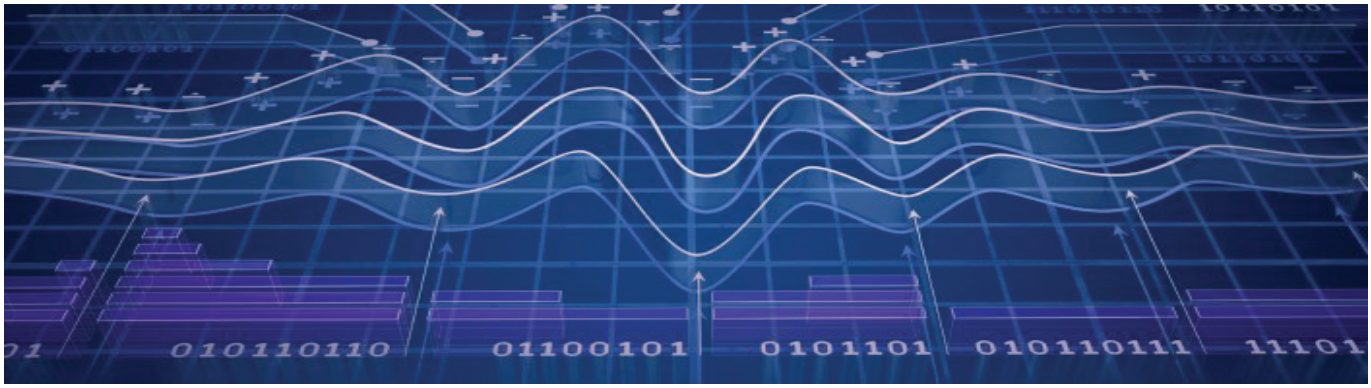
Figure 1. A Balanced Approach

Statistical Modeling

The statistical modeling starts by profiling network entities' historical activities. An example of a user profile is a histogram distribution of user's login counts to a set of devices; an example of a device profile is a distribution of volume of bytes copied over a set of users. One of Exabeam's outlier analysis tools is based on p-value for statistical hypothesis testing to flag whether the current activity is an anomaly against the profile. If so, an alert from this particular anomaly is assigned an a-priori expert-assigned score, then adjusted based on past triggering patterns. The expert-assigned scores reflect our security researchers' knowledge about the importance, while an example of the score adjustment considers whether this triggered alert is a frequent event, for better false-positive control. Sessions with the highest accumulated scores from the triggered alerts are presented to analysts, with easily-understood reasons for the high scores.

Profiled data types can be either categorical or numerical in nature. An example of categorical data is tracking login counts for each asset to which a user has connected. An example of continuous numerical data is the number of bytes transferred from a device. Profiling for continuous numerical data has non-trivial implementations. One of the implementations we use is to organize or group numerical data points to a dynamic histogram with distinct clusters or bins.

To dynamically construct a histogram of numerical data, we use an unsupervised clustering algorithm. It starts by placing each point into a single group, iteratively merging the two closest groups until convergence. The criteria for evaluating the clustering quality is based on silhouette coefficient, which measures consistency within clusters of data. This clustering step must be done periodically to adapt to new data, to ensure the fidelity of clusters.



Built profiles are further conditioned with a series of steps to ensure their highest fidelity. An example is the convergence check; profiles not meeting certain criteria are not used. Another example is delayed training – a step used to ensure that likely malicious events do not pollute the profiles built to model normal behaviors. Another example is noise removal; popular but uninteresting events need not be part of the profiles or they become dominating noise in the model.

Selecting the right analytics tool for activity profiling is only part of the equation. A challenge even before the use of analytics is in knowing, among the endless possibilities, which features to engineer, what statistical entities to track and compute, and how to construct uncorrelated statistical indicators from the ground up. The designs and choices of the right statistical indicators or profiles are built upon the many years of experience our in-house security experts have gathered. The next challenge is in the actual implementation of analytics that scales, particularly when algorithms need to learn and score in real time. With hundreds of thousands of events streaming into the system every second, moving and shuffling the events to compute the statistics and counts while being subject to memory constraints is a real platform engineering challenge. This requires space-time tradeoff. A description of software engineering methods involved in implementing this and other similar algorithms of the Exabeam product is beyond the scope of this paper.

Machine Learning

Machine learning is a method that is used to devise complex models and algorithms for the purpose of learning or making predictions from data. In the UEBA context, using a single complex modeling technique to detect anomalous user behavior has low likelihood of success. Lack of ground truth (known breaches) and the use of unsupervised learning doesn't bode well for UEBA applications that require very low false positives. Enterprise environments are complex, fraught with lots of uncertainties and ill-defined data sets. Network context information is not always reliable. User behaviors are not necessarily boxed in and the environment is always in a state of flux. All of these factors make it difficult to materialize a monolithic detection algorithm over multiple data sources. Within UEBA, machine learning shines in the areas of context estimation and targeted detection use cases. Machine learning solutions in these areas provide capacity to statistical analysis to reduce false positives and increase precision. Below are examples of patent-pending machine learning applications used in Exabeam's UEBA system.

Service Account Classification

If we see an account performing a high volume of activity that might be abnormal for a human user but perfectly normal if the account is a service account. Raising an alert without considering the context is prone to a high rate of false positives. Therefore, wherever possible, Exabeam leverages an enterprise's existing account labeling information for the UEBA system deployment. However, not all environments have such data readily available; more often than not, the information may be incomplete since such data is hard to maintain and it mushrooms out of IT control as the environment grows. Also, maintaining such data typically has not been critical for core IT operations. Despite the labeling imperfection, Exabeam has created and deployed several algorithms to estimate or even correct the labeling information of an account, whether it is a service account or a user account.

One patent-pending method leverages information from Lightweight Directory Access Protocol (LDAP) files that enterprises maintain for directory services to provide records of network entities such as users and assets. Every entity is described by a collection of key value pairs. Some keys are semi-standardized, some are not, and the value of a key might be free text. Human eyes tend to do a fair job in identifying whether an account is a service or user account by reading the key-value pairs. How do we create an algorithm for a computer to do the same or better for automated classification? We first represent each account with a binary vector of size N where N is the number of keys used in the LDAP files; N is typically in the order of several hundred across enterprise environments. Each dimension holds a Boolean value indicating whether a key is present in the account or not.

Framing the problem in this way, with known user or service account labels, we can readily build a supervised classification model to output a prediction score to classify new unknown accounts in the N-dimensional space. Figure 2 shows the resulting receiver operating characteristic or ROC curve in predicting service accounts.

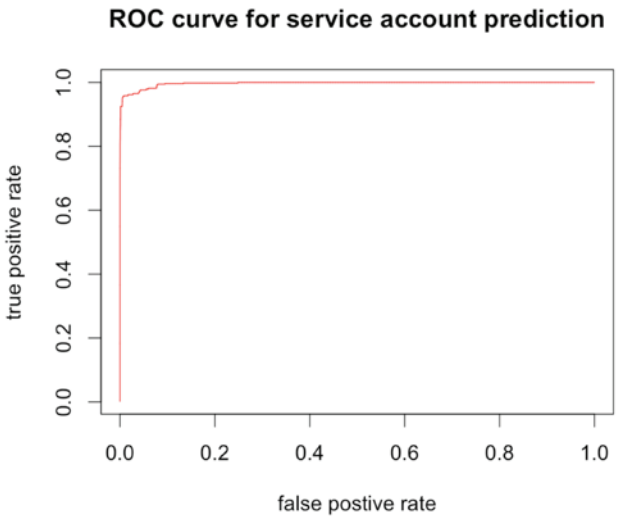


Figure 2. Service Account Prediction ROC Curve

There are still other ways to skin the classification problem. The text-based account classification described above is one. Another is to classify accounts using their behavior data. We first define behavior features derived from accounts' activities recorded in the logs. Obvious behavior features include number of events generated or received by the accounts, number of hosts an account is connected to, etc. Assuming most of the account population are user accounts, then given the feature collected, an unsupervised learning approach such as one-class Support Vector Machine (SVM) or an algorithm based on dynamic thresholding is used for the purpose of account classification.

Asset Type Classification

Another important area of context is whether the user's accessed machine asset is a workstation or a server. This classification is critical to enable Stateful User Tracking™ in UEBA to organize activities in user sessions. Since a meaningful user session starts with a user logon event to a workstation among other conditions, we need to know the logon machine's type. Similar to service accounts, assets proliferate and mushroom within the environment, and usually there isn't a central repository that categorizes the different types of assets. Even when such a repository is present, it is rarely up to date. This presents a good application of machine learning: build a classifier using the asset's own behavior data. The activity stream from a workstation is different from that from a server. Designing a set of behavior indicators and applying a relevant learning algorithm allows us to build a classifier for the purpose of identifying the type of an asset on the network.

Threat Detection

Another use case for machine learning is targeted threat detection. There is no single magical algorithm that processes multiple data sources to find noteworthy outliers. In enterprise logs, data sources are heterogeneous; data may be incomplete; logged entities' behaviors are nuanced and require expert analysis. Therefore, it's best to use machine learning to address targeted use cases. An example is the detection of algorithmically-generated domain names.

It is a common practice for malware to establish communications over a pseudo-random domain name that is generated by an algorithm. Access to such a domain is indicative of malware communicating outside of the network. The pseudo-random domain names are impossible to detect using regular expressions but can be detected through probabilistic language modeling. Here we might use a letter-based N-grams model to determine the likelihoods of N-letter sequences learned from a large corpus of normal-looking words or web domains. For example, if N=2, bigrams from the word "exabeam" are "e-x", "x-a", "a-b", ... "a-m". We have about 700+ such bigrams and we can train their likelihoods from millions of domain names. The model represents how a large collection of normal domains appears. Given a pseudo-random domain name and its collection of bigrams, it will score low against the model. Hence, the domain is likely algorithmically-generated and can be further evaluated with more contextual information, such as when the domain was first registered.

Peer Context Derivation

If we have an alert that a user accessed an asset for the first time, how much weight should we give to this alert? The alert must be viewed within its context. A good context is whether members of his peer group have or have not accessed this particular asset before. Active Directory (AD) data does provide some of a user's peer group

context information, albeit incomplete or out of date in typical enterprises. Peer group labels such as department, title, or office location have been observed to have far less than an ideal 100% coverage of a user population. Lacking that complete labeling coverage, use of peer group for alerts' context is suboptimal.

On the other hand, the machine learning-based recommendation system is well-suited for this problem. Netflix, Amazon, and others in the data analytics industry have long used recommendation system technology to predict the next movie or items a user is likely to buy, based on data from other members who have shared the same buying patterns. By the same token, we can use a recommendation-type system to find a user's peer group if they share the same historical asset access behavior patterns. Given a first-time asset access alert for a user, we can keep or remove the alert based on the frequency of his peers' access to the same asset. This is an example of using machine learning to reduce false positives. We don't always have to focus on the detection use cases for UBA. Reducing false positives via machine learning increases the precision rate; hence, better detection.

Conclusion

The paper illustrates Exabeam's beliefs and approaches to data analytics in UEBA including core statistical analysis infused with security knowledge, and a system design that is easy to configure and use, with easily-explained results. The power of machine learning is harnessed for well-chosen use cases to provide targeted detection or contexts. As-is, the system is fully extensible to accommodate new security analytics use cases for different log sources, whether AD, GitHub or cloud access log. In addition, the system easily provides hooks for customers' data scientists to plug in their externally-derived results to integrate as contextual data for constructing new statistical indicators to further enhance threat detection. As the security climate evolves, Exabeam will continue to advance in both analytics and platform to meet customer needs.

For more information, please visit the Exabeam website at http://www.exabeam.com, or send an email to **info@exabeam.com.**