2020 IEEE International Conference on Big Data (Big Data) | 978-1-7281-6251-5/20/$31.00 ©2020 IEEE | DOI: 10.1109/BigData50022.2020.9378365

# BEAM: An Anomaly-Based Threat Detection System for Enterprise Multi-Domain Data

Derek Lin
*Exabeam Inc.*
*Foster City, California, USA*

Anying Li
*Exabeam Inc.*
*Foster City, California, USA*

Ryan Foltz
*Exabeam Inc.*
*Foster City, California, USA*

*Abstract*—**Organizations are faced with the ever-increasing risk of security threats. Security threats are multifaceted and present different levels of challenges to the defenders. While traditional deterministic signature and correlation-based methods serve limited purpose, behavior-based anomaly detection methods are best suited for identifying signature-less threats such as those from external adversary or insider activities. Challenges in building an anomaly detection system for enterprise security are numerous. This paper introduces a new anomaly detection system that addresses the feature engineering process for multi-domain data in enterprises and provides a Bayes-based risk scoring method for information fusion. The system is adaptive to dynamic user and network behaviors and produces interpretable outcomes by design. To perform quantitative evaluation against a baseline system without known labeled attack data, we propose a method to synthesize the ground truth to demonstrate the improvement in detection performance.**

## 1. Introduction

Organizations are faced with ever-increasing risks from security threats. According to a recent Verizon report [1], seventy percent of cyberattacks were perpetrated by outsiders, while thirty percent of them involve insiders. Insider threats are particularly hard to guard against, whether they're from someone using their legitimate access to data for nefarious purposes or conducting network activities in ways that violate acceptable user policies. Since the activities from an insider or an external adversary are dynamic, traditional means of detecting threats that match data against known blacklisted signatures or hand-crafted correlation rules are largely ineffective or irrelevant. In recent years behavior-based anomaly detection works have emerged to address these security threats; in particular, user and entity behavior analytics (UEBA)[2] has emerged in the cybersecurity industry as a viable approach to detecting behavior anomalies.

Enterprise security products generate a large variety and volume of entity or user activity event data of different categories or ”domains”. Example activity domains include asset logon and access, account management, VPN, email, proxy/firewall, cloud application, endpoint, and physical access. The events processed by and stored in security information and event management (SIEM) systems are leveraged for behavior modeling in order to flag deviations for alerts. Challenges in building a behavior-based anomaly detection system are multifold. They include a method to derive features for continuous and adaptive behavior profiling, the strategy of information fusing behavior anomalies from disparate data domains, the need of near real-time processing, the requirement for self-explanatory outcomes, and the reality that volume of labeled ground truth for malicious events is practically non-existent for conducting quantitative evaluation of a built system.

This paper presents a *BE*havior-based *A*daptive *M*ulti-domain threat detection system, called BEAM, to detect security threats in enterprise network and user events in real-time. The feature engineering process is principled and made practical across diverse data domains. The Bayesian method is used to assess risks in prioritizing user session risks. The system is self-learning by updating itself from the continuous user activities. The outcomes are self-explanatory, as this is a white-box system. For evaluation we introduce a method to synthesize labeled ground truth for quantitative assessment, as well as compare BEAM against another baseline system that relies on carefully crafted rules and manual score tuning.

The outline of the paper follows. Section 2 provides background and related work. Section 3 describes the system from the feature derivation process to the Bayes-based risk scoring mechanism. Section 4 describes the data sets, evaluation procedures, and experimental results. We conclude in section 5.

## 2. Related Work

Security threat detection systems leveraging a behavior-based approach use data from a variety of domains to detect malicious activities. Although commercially available systems [3][4][5] exist, behavior-based threat detection remains a challenging and largely unsolved problem. Past academic work from Eldardiry et al, [6] finds anomalous users exhibiting inconsistent behavior across the data domains. It predicts a user's latent peer cluster index in one domain from the cluster indices in all other domains. Instead of anomalies for users, our work identifies anomalies for individual user sessions, more suitable for security center operations. Bose, et al., [7] details an architecture for Spark streaming-based anomaly detection. It builds features for all domains at once and uses the k-nearest neighbor algorithm for user-based and role-based anomaly detection. The work [8] is also a stream-based system that employs clustering in order to find local outliers. Zargar. et al., [9] takes a knowledge-agnostic approach on data entries. The main hypothesis is to alert on a user if they show a behavior, such as the presence of new data entries unknown to their own history, that is exclusive to another user in a different data channel. The above systems do not consider the continuous learning from user feedback.

Veeramachaneni. et al., [10] extracts features via deep learning and matrix decomposition before applying supervised learning for classification. Tuor, et al.,[11] uses variants of deep neural networks (DNNs) and recurrent neural networks (RNNs) to recognize activity that is uncharacteristic of each network user. Liu. et al., [12] uses deep autoencoder to detect the insider threat. Yen. et al., [13] applies K-means after feature extraction via principal component analysis (PCA) to find outlying hosts. Shyu. et al., [14] similarly uses PCA for feature extraction for outlier detection. Black-box systems such as these inevitably lack the interpretability for output alerts - a critical requirement for actionable outcomes.

Maloof. et al., [15] uses a three-level, tree-structured Bayesian network consisting of Boolean random variables for features. Gheyas [16] compares a number of insider threat detection systems and shows the research community has a preference towards a Bayesian-based learning algorithm. We adopt the Bayes method because of its many advantages.

## 3. BEAM System Design

BEAM is an automated behavior-based anomaly detection system for enterprise-level security threat detection. Factors influencing its design decisions are as follows:

- The system outcome must be self-explainable. This is a critical requirement for security operation centers. Actionable security alerts start with the understanding of the alerts themselves.
- Enterprise-level applications require near-time operation. Feature extraction must be computationally efficient at the event level for real-time or near real-time alerting.
- It's not uncommon for enterprises to add new data domains for monitoring. Analysts demand the flexibility of adding new anomaly indicators to the system. Behavior modeling framework must be flexible to accommodate to both data domain additions and feature changes.
- The model must adapt to the dynamic nature of network and user behaviors. User, peer, and organization-level profiles are continuously updated.
- Whenever expert or analyst feedback is available, the system must leverage it and allow a path to incorporate the human inputs to update the model parameters.

BEAM has been designed to meet the above requirements. At the high-level, during its training stage, network entities are individually profiled with their activities tracked across different data domains. (A network entity is a user, a user's peer group, or the entire organization.) Learning features are defined per network event. They include a rich set of anomaly detectors and the associated contexts. A Bayesian network is thereby learned from these features. During the scoring stage, each user event is assigned a risk score based on its observed feature values. Throughout a given day, a user's overall risk is aggregated from the risks of all their seen events so far. At any time, a list of score-ranked users is available for investigation. We next describe BEAM in more details.

### 3.1. Data Preprocessing

Enterprise networks and services are supported by a wide array of security, network, and server infrastructure products. Example data sources supporting user activity domains are:

- **Active Directory service from Microsoft**: authentication and authorization events for all users and computers in a Windows domain network
- **Identity management products**: events for user or computer account creation and deletion
- **Web proxy products**: web activity events and alerts, such as blocking connections to potentially malicious domains or URLs
- **Data loss prevention products**: security events for alerts relating to data exfiltration
- **VPN products**: virtual private network login events such as login success or failure
- **Data security products**: events for database activities, such as database queries
- **Badging system products**: events for physical building or gate access
- **Cloud services**: events or alerts for cloud-based application usage

Raw events of the data domain may be from multiple vendors' products or from a single vendor's products (but of different versions). It's critical to parse and normalize the data sources so that events of the same domain–but having varying formats–are merged and normalized for analytic purposes. For example, to normalize events for the domain of user asset logon and access, a Kerberos logon event is normalized by merging the Windows AD event IDs 4768 and 672 [17]; a remote-access logon event is for Windows event ID 4769 or 2624 with logon type 3 or 8; an account-creation meta event is for Windows event ID 4720 or 624. Data parsing and normalization in security log data requires domain expertise and is a non-trivial, critical step prior to modeling. (Hereafter we refer to *event* as the normalized event.) Denote domains as $C_v$ in which $v$ is the domain index, $1 <= v <= V$ where $V$ is the total number of domains.

## 3.2. Risk Framework

BEAM has the real-time operational requirement of per-event scoring of cumulative risk of user activities since the start of a day. BEAM defines a user activity session $S$ to start from the beginning of a 24-hour period until the current time, but not beyond the end of the 24-hour period. A session then consists of a collection of $T$ user's observable events $\{e_t^v\}$ where $t$ is the event index and $v$ is the domain to which the event belongs.

We define the overall session risk as the sum of domain risks $R_{C_v}$ for all observed domains in a session.

$$R_{total} = \sum_{C_v} R_{C_v} \quad (1)$$

where a domain risk is the sum of the individual event-level risks $\{R_{e_t}^v\}$ and the session-level risk $R_s^v$ from the domain.

$$R_{C_v} = \sum_{t}^{T} R_{e_t}^v + \alpha R_s^v \quad (2)$$

where $alpha$ allows some flexibility tuning of the relative contributions of event-level and session-level risk.

### 3.2.1. Event-level features.

The risk $R_{e_t}^v$ is a Bayes risk assessed based on a vector of features derived from the event $e_t$ of the domain $C_v$. Features of an event are a collection of *anomaly detectors* and *context*. An anomaly detector indicates whether an aspect or a data element of the event is unusual with respect to a tracked behavior profile. A profile is a historical description of the tracked data that is continuously updated.

**Anomaly detector feature:** Where categorical data is involved, the profile is time-based, with the most recently accessed times tracked for the values of data. For example, a profile tracking a user's accessed assets consists of a list of tuples of an asset and the most recent timestamp accessed. Where numerical data is involved, the profile is a description over past observed values via the boxplot method or a parametric probability function. For example, a profile tracking the the size of email a user sent uses boxplot to describe the interquartile range of the sample. Behaviors tracked by profiles are simple to understand and are useful for interpretability. BEAM builds a comprehensive set of behavior profiles for all domains per user, as well as per user's peer group and per the entire organization. A user may belong to multiple peer groups–for examples, department, title, geo-location, and security access group they are associated with.

An anomaly detector checks whether a current data item is considered new, unusual, or an outlier in accordance with a profile. In the case of categorical data, an anomaly detector returns TRUE if the data is new to the profile or was last seen more than $N$ days ago ($N = 30$ has worked well in practice). However, if the data in question is new in the population – for example, in the case where the user accessed for the first time a newly deployed asset in an enterprise – the detector doesn't consider this be an anomaly. The data must be matured or "converged" before it's evaluated. We say the data has converged if it has aged more than $M$ days. ($M = 5$ has worked well in practice.)

In the case of numerical data, the detector returns TRUE if they are considered as an outlier that exceeds a certain threshold in the profiled distribution.

**Context feature:** In addition to the anomaly detectors, static and factual context data serve as features of an event for risk scoring. Context data is available from data sources such as an LDAP database or other IT operation databases; examples include a user's privileged access status over certain assets or whether an asset is a workstation or a server.

In assessing the risk of an event, the role of context features is to provide context to calibrate degrees of risk due to observing the unusual events detected by the anomaly detectors. For example, the risk due to an anomaly detector monitoring a user who is running an unusual process is best calibrated by whether the machine running the process is run on is a critical server or a workstation.

### 3.2.2. Session-level features.

The risk $R_s^v$ in Eq. 2 is a Bayes risk assessed on a feature vector derived from events of domain $C_v$ in a session. Similar to event-level features, the session-level features are a collection of anomaly detectors and context indicators. Events from a session are aggregated to build profiles and detectors; for example, the number of bytes transferred in this session, or number of assets accessed in this session.

### 3.2.3. Bayes modeling.

Let $R$ denote either $R_{e_t}^v$ or $R_s^v$. During training, the goal is to learn $R$ for each user's every possible feature vector $\vec{f} = (f_1, f_2, ., f_I)$ of length $I$. During evaluation, $R$ is simply looked up based on the current feature vector; the user's overall session risk then follows as in Eq. 1.

We define $R$ as a user's posterior probability that the data is malicious or is an anomaly of high interest given the feature vector. Denote $M$ for "malice or anomaly of high interest" and $L$ for "legitimate or of no interest".

$$R := P(M|\vec{f}) = \frac{P(\vec{f}|M)P(M)}{P(\vec{f}|M)P(M) + P(\vec{f}|L)P(L)} \quad (3)$$

To compute $P(\vec{f}|L)$, we assume the observed historical data is largely normal and legitimate, and that the volume of any present malicious or anomalous events is both negligible and statistically insignificant for the purpose of our risk inference.

The data used to compute a user's $P(\vec{f}|L)$ is their local sufficient statistics smoothed by the population's global sufficient statistics:

$$P(\vec{f}|L) = \frac{(1-\beta) * C_0 * Count_u(\vec{f}) + \beta * Count_g(\vec{f})}{(1-\beta) * C_0 * \sum_{\vec{f'}} Count_u(\vec{f'}) + \beta * \sum_{\vec{f'}} Count_g(\vec{f'})}$$
$$where\ C_0 = \frac{\sum_{\vec{f'}} Count_g(\vec{f'})}{\sum_{\vec{f'}} Count_u(\vec{f'})} \quad (4)$$

$Count_u(\vec{f})$ is the count of sessions from a user's own historical sessions in which the feature vector $\vec{f}$ is observed. $\sum_{\vec{f'}} Count_u(\vec{f'})$ is the sum of all such counts from all possible observed feature vectors within the same user's history. Similarly, $Count_g(\vec{f})$ is the count of sessions from all users in which the feature vector is observed, and $\sum_{\vec{f'}} Count_g(\vec{f'})$ is the sum of all such counts from all possible observed feature vectors across the user population. $C_0$ is a scaling factor so that the mass of local sufficient statistics has 1-to-1 equality with that of the global sufficient statistics. $\beta$ is a parameter between 0 and 1 controlling the smoothing of the population-global vs. user-local sufficient statistics.

The likelihood $P(\vec{f}|M)$ is assumed to be uniform across all observed feature vectors, unless otherwise preferred by expert assignment or by analyst feedback. Indeed, the Bayes inference scheme allows expert opinion to adjust it and user feedback to update $P(\vec{f}|M)$ in the production setting. In the absence of expert knowledge or analyst feedback, the risk in Eq. 3 simply reflects the degree or rarity or anomalousness of the feature vector in predicting $M$.

Lastly, the priors $P(L)$ and $P(M)$ in Eq. 3 are assumed to be 0.5, with no previous assumption pertaining $L$ and $M$; this is also a practice in Bayes-based spam email detection software [18].

**3.2.4. Independence of domains.** The overall session risk is the sum of domain risks in a session, as in Eq. 1. Since the overall risk is additive, this presumes activities on domains are independent. For example, we believe the user behaviors on VPN data domain is reasonably independent from the file-access data domain. This assumption improves the "modularity" in the feature design as such feature changes are localized. It also improves the scalability as new domains are added over time. Table 1 shows example domains this work has analyzed.

**3.2.5. Conditional independence of feature groups in a domain.** To compute the risk $R_{C_v}$ from the domain $C_v$ given its feature $\vec{f}$, the likelihood $P(\vec{f}|L)$ must be evaluated. While it's possible to calculate this likelihood directly regardless of the size of $\vec{f}$, in practice, the conditional independence of features is best exploited, such that:

$$P(\vec{f}|L) = \prod_{j=1}^{J} P(\vec{g}_j|L)$$

| **Domain** | **Event count** | **User count** |
|---|---|---|
| Account login and access | 1916464 | 2112 |
| Account management | 109 | 7 |
| Application | 114866 | 1664 |
| Badge access | 12 | 5 |
| Data loss prevention | 45482 | 432 |
| Database | 15724371 | 21 |
| Email | 42950 | 414 |
| Endpoint | 22497648 | 1019 |
| File access | 914175 | 704 |
| Network access and aogon | 1241288 | 1999 |
| Vendors' alerts | 367 | 37 |
| VPN | 378 | 196 |

TABLE 1: An enterprise environment

where there are $J$ conditional independent groups $\vec{g}_j$, each consisting of a non-overlapping subset of features from $\vec{f}$. Table 2 shows an example of assigning event-level features for the VPN domain to four conditionally independent groups.

| Group | Event-Level Features |
|---|---|
| Group 1 | Anomalous VPN realm for user<br>Anomalous VPN realm for user's peer<br>Anomalous destination host for user<br>Aomalous destination host for peer group |
| Group 2 | Aomalous source host for user<br>Anomalous source host for organization<br>Anomalous OS for user<br>Anomalous source IP for user |
| Group 3 | Anomalous VPN activity for user<br>Anomalous VPN activity for peer group |
| Group 4 | Failed login<br>Source IP is on blacklist<br>User is a contractor or a partner<br>Account is disabled |

TABLE 2: Conditional independent feature groups of VPN domain

The benefits of organizing features into independent groups are two-fold. First, the estimation for the joint likelihood is expected to be more robust. The total likelihood of joint features is the product of likelihoods of conditionally independent groups of features; each group has a manageable or smaller number of features, reducing the data fragmentation issue that comes with jointly estimating the likelihood of a large number of features.

More importantly, structuring the features to conditionally independent groups improves the interpretability of the outcome. Given a high scoring session, rather than presenting the entire feature vector $\vec{f}$ where size $I$ may be large, the system can selectively present only the critical feature groups of smaller size that significantly contributed to the session score. This improves the readability of the outcome – a critical consideration in a production system.

BEAM organizes anomaly detector features and context features to groups whenever possible. While the group design is a manual process, there are some guidelines to follow. For example:

- Anomaly detectors using profiles built on similar data are grouped together, since they're not expected to be independent. For example, detectors using profiles relating to a user's source machine are grouped together. Similarly, anomaly detectors relating to a destination machine to which the user activity is intended are grouped together

- Anomaly detectors for a user, their peer, and the organization regarding similar activity are grouped together, since they're not independent. For example, if a current user event is anomalous to that user's departmental profile, it must be anomalous for the user itself, though not necessarily vice versa.
- A context feature is part of anomaly detector group if the context is useful to calibrate the risk from the triggered anomaly detectors in the group. For example, the context of whether a user is an administrator is grouped with various anomaly detectors that checks for the unusual access to an asset from a user, their peer, or the organization. The risk from the detectors depends on whether the user is an administrator or not.

**3.2.6. Risk-ranking enhancement.** The session risk in equation 3 assumes all events independently contribute risk to a session, regardless of the domains they come from. Some domains, or a combination thereof, are more infrequently observed than the others. Sessions with events from rarer domains, or domain combination, intuitively should be ranked higher than others. It's then useful to introduce a weighting factor on session risk $R_{total}$ to reflect the ranking preference based on the domain diversity observed in a session. Similar to the muliti-view fusion work from Zhou. et al., [19], we regard each domain as a view into session activity. We introduce the domain diversity factor $W_D$, and the corresponding weighted risk score $R^D_{total}$:

$$W_D = \frac{\prod_v^V P(C_v)}{P(C_1, C_2, ..C_V)}$$

$$R^D_{total} = R_{total} * W_D \tag{5}$$

where $C_v$ is the $v$'th domain in which any of the anomaly detector features from the domain have returned TRUE in a session. $W_D$ has the desired property that, when some domains are dependent to one another, it will be less than 1.0, thereby penalizing the session score. Otherwise when domains are truly independent, such that they provide maximum information in a session, $W_D$ will be 1.0, thereby preserving the session score.

# 4. Experiments

## 4.1. Baseline System

BEAM is compared with a baseline threat detection system deployed in production. The latter consists of a large collection of rules of size $N$, against which activity events are evaluated. Its $i$'th rule $rule_i$ is individually crafted and has a manually assigned score $S_i$ upon triggering. The total risk of a user session $S_{total}$ is simply the sum of scores of triggered rules:

$$S_{total} = \sum_i^N S_i \tag{6}$$

A rule is either fact-based or anomaly-based. A fact-based rule simply checks whether the data deterministically matches a certain fact; for example, whether the IP address of the event is on a blacklist. Similar to BEAM's anomaly detector, an anomaly-based rule evaluates the data against a user profile. For categorical data, the profiles are count-based histograms. For numerical data, the profiles are constructed by first clustering the data into bins of numerical ranges, on which count-based histograms are built over the bins. Given the histogram-based distributions, both event-level and session-level rules are defined and enabled to trigger if the data value is new or unusual to the profiles based on its p-value against some threshold. Finally, a rule may have various, manually crafted conditions as part of its triggering requirements; for example, data must also have a certain context, or the rule can only trigger on condition of other rule's triggering status. Table 3 shows rules counts of the baseline system per data domains.

The baseline system performs satisfactorily in production but does require significant amount of development and tuning efforts specific to an environment. The hand-assigned rule scores are based on experts' subjective opinions. Rule definitions sometimes need complicated and overlapping conditions to ensure the precision. Tuning of various internal parameters, including the individual p-value thresholds used to trigger anomaly-based rules, is non-trivial. Furthermore, rules are independently developed as needed; each is specific in scope and does not generalize. As a result, the number of rules grow over time and the statistical independence of them is not assured. The summation of scores from triggered rules to comprise the session score is not optimal.

## 4.2. Data Setup

We evaluated BEAM on a corporate network environment with events from multiple data sources. The environment has total 4,882 total users, 120,667 user sessions, and 43,098,903 events. Table 1 breaks down the volume of per-domain events observed. Sixty-three days of data are available for experimentation. The first month is used to initialize profiles, which are then continuously updated throughout the rest of the data period. The second month is used to collect the sufficient statistics from which anomaly detectors will build $P(\vec{f}|L)$. The remaining two days are used for scoring and evaluation.

**4.2.1. BEAM features.** As discussed in the Section 3.2, event-level features and session-level features are designed for each domain. Conditional independent feature groups are defined. For feature design, we capitalize the information from rules in the baseline system. We effectively translate and normalize the rules to the learning features, but with

TABLE 3: Comparison of Baseline rules vs. BEAM features across domains

| Domain | #Baseline rules | BEAM Event-Level Features | | | BEAM Session-Level Features | | |
|---|---|---|---|---|---|---|---|
| | | #anomaly detectors | #context | #feature groups | #anomaly detectors | #context | #feature groups |
| Account login and access | 47 | 2 | 5 | 2 | 6 | 0 | 2 |
| Account management | 34 | 8 | 3 | 5 | 0 | 0 | 0 |
| Application | 33 | 10 | 2 | 7 | 1 | 0 | 1 |
| Badge access | 16 | 3 | 2 | 2 | 2 | 3 | 1 |
| Data loss prevention | 30 | 13 | 0 | 3 | 1 | 0 | 1 |
| Database | 21 | 11 | 0 | 2 | 2 | 0 | 1 |
| Email | 20 | 4 | 5 | 2 | 3 | 0 | 1 |
| Endpoint | 14 | 6 | 5 | 2 | 3 | 0 | 1 |
| File access | 17 | 8 | 1 | 4 | 5 | 0 | 1 |
| Network access and logon | 46 | 10 | 0 | 3 | 2 | 0 | 1 |
| Security alerts | 3 | 2 | 1 | 1 | 0 | 0 | 0 |
| VPN | 20 | 10 | 5 | 4 | 3 | 0 | 1 |

no new net information added. Table 3 shows the number of features and conditionally independent feature groups designed for BEAM for all evaluated domains. The number of features is smaller than that of rules in a domain; sometimes this difference in numbers is dramatic. For example, only seven BEAM features are used to completely capture the 47 rules relating to the asset logon and access domain. There is less complexity involved in configuring BEAM's features than rules.

Despite the more compact information representation of the features, the combinatorics of features is more expressive than the rules – sometimes capturing data conditions not covered by the baseline rules. For example, in activities from the asset logon and access domain, it's common to have the following two rules: the user rule with an anomaly detector that checks if their user's asset access is unusual, and the user's peer rule with an anomaly detector that checks if this asset access is unusual for their peer group. These two rules alone don't capture the scenario where the user has an unusual asset access but not for his peer. In BEAM, all four activity scenarios resulting from the two anomaly detectors (each outputs a Boolean state) have scores, thus providing a full risk scenario coverage.

**4.2.2. Synthesized ground truth.** Unless explicit penetration testing exercises are conducted, ground truth labels for known security threat data are practically non-existent. For the purpose of evaluating anomaly-based threat detection systems, we synthesize the ground truth as follows. Two lists of user sessions are randomly sampled from the evaluation period; they don't overlap in users. Sessions from the first list are the $victim$ sessions; sessions from the second list are the $attacker$ sessions. Victim and attack sessions are randomly paired. A victim session is spoofed by injecting events belonging to the attacker session. Such spoofed victim sessions are the synthesized, positive-labelled sessions containing anomalous activities not known to the victim user. We use such spoofed data sets for evaluating the relative performance of BEAM and the baseline systems. The proposition is that the more successful, anomaly-based threat detection system should separate the spoofed sessions from the non-spoofed sessions better.

The scope of the evaluation can be refined and made more targeted by varying the spoofing conditions. We created two sets of synthesized ground truths for two different attack scenarios.

- **External attack**: attack events are injected into a victim session without modification. Since source devices or IPs aren't modified, the attack data looks as if they came from an external adversary.
- **Insider attack**: attack events are modified to match the dominant source device, source IP, destination device, and browser's user agent string in the victim's history. Since the attack activity events look as if they came from the victim's normal hardware, the anomalous events appear to come from the victim user himself, thereby mimicking insider activity.

We evaluate the accuracy performance on these two scenarios. Each scenario has 4,533 negative-labeled legitimate sessions and 1,071 positive-labeled attack sessions for the two-day evaluation period.

Note that this ground truth data construction method shouldn't be used to establish accuracy reports for operational purposes. Rather, it's intended only to evaluate the relative performance difference of anomaly detection systems.

## 4.3. Results and Analysis

BEAM and the baseline system are evaluated on both the external and insider attack scenarios. Figure 1a shows the accuracy performance in receiver operating characteristic curve (ROC) for the external attack detection scenario. Figure 1b shows a zoomed-in view in the low, false-positive-rate region, where a suitable threshold is chosen for operation. (This rate is the ratio of the number of flagged sessions over the number of sessions with non-zero session scores.) For example, over a two-day testing period, there are 2,497 sessions in the baseline run with non-zero scores. The 2% false positive corresponds to about 25 flagged sessions per day.

BEAM performs significantly better than the baseline in the external attack detection scenario. At the 2% false positive rate, BEAM is 40% better in the attack detection rate than the baseline. Figure 1c shows the similar comparison but for the insider attack detection scenario. At the 2% false positive rate, the accuracy advantage of BEAM continues to hold for this latter scenario.

The parameter $\beta$ in Eq. 4 controls the smoothing of a user's local sufficient statistics with the population's global sufficient statistics in computing a user's $P(\vec{f}|L)$. Figure 2 shows the effect of changing this parameter in a zoomed-in ROC plot. We observe that $\beta = 1$ gives the best performance, where the local and the global sufficient statistics are one-to-one equal in mass. In addition, relying on the local statistics alone without smoothing, i.e., when $\beta = 0$, doesn't perform well since not all users have the historical data volume to reliably train their $P(\vec{f}|L)$.

Operationally, the diversity factor $W_D$ facilitates the security operation center preference that sessions with rarer combinations of domains should be ranked higher than others. To show its effect as a correlated consequence in this application, sessions involving more domains with triggered anomalies are expected to rank higher. Among those ranked in the top 50, Figure 3 shows the histogram of session counts over the number of domains with present anomalies in a session. Among these sessions, we observe the diversity factor indeed results in more sessions having a higher number of domains.

Figure 4 shows the accuracy performance of the diversity factor in a zoomed-in ROC plot. While $R^D_{total}$ expresses the desired session ranking preference, it also has a slight advantage in improving the accuracy performance of $R_{total}$. An alternative diversity factor definition to 5 is the probability that the session is malicious given the number of domains present, or $V$, in the session

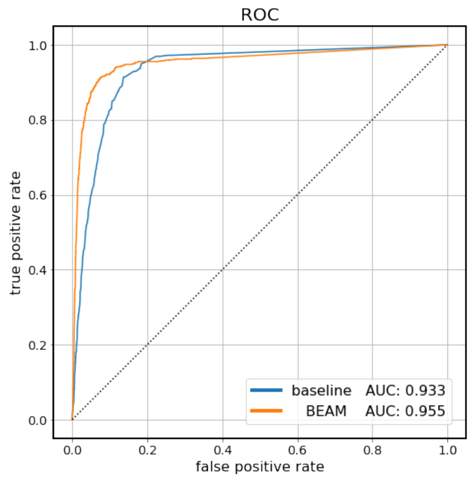
$$W_D^{'} = P(M|V)$$

$$R^{D'}_{total} = R_{total} * W'_D$$

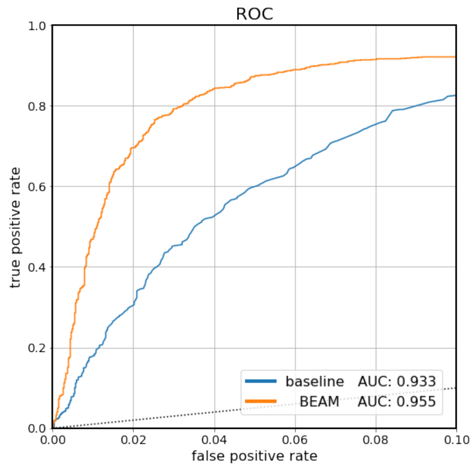
While $R^{D'}_{total}$ has the same intended effect as $R^D_{total}$, $R^{D'}_{total}$ underperforms in accuracy, since it doesn't take the domains themselves into account.

## 4.4. Conclusion

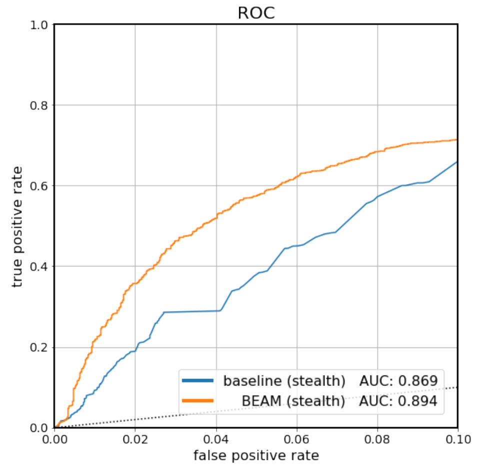
We introduced a behavior-based anomaly detection system to detect threats from enterprise network data. In contrast to common threat detection systems that rely on hand-crafted rules and expert-determined scoring, BEAM eliminates the complexities in developing and manually tuning the rules. We outlined its feature engineering process and introduced a Bayes-based risk scoring framework. This framework easily accommodates experts' preferences and analysts' labeling feedback in updating risks. Due to its white-box nature, the system outcome is highly interpretable. Finally, we demonstrated BEAM performs better in accuracy than a conventional rule-driven threat detection system.



(a) external attack scenario ROC



(b) external attack scenario ROC, zoomed-in view



(c) insider attack scenario ROC, zoomed-in view

Figure 1: Baseline vs. BEAM ROCs

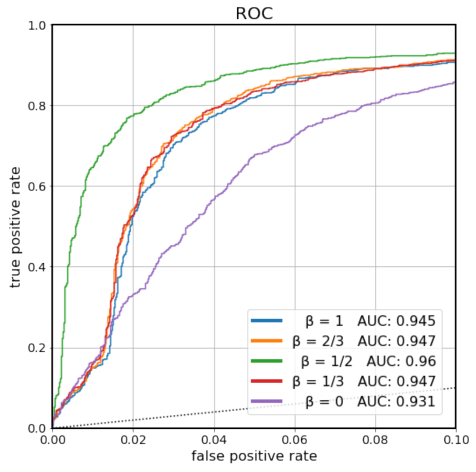


Figure 2: Effect of $\beta$

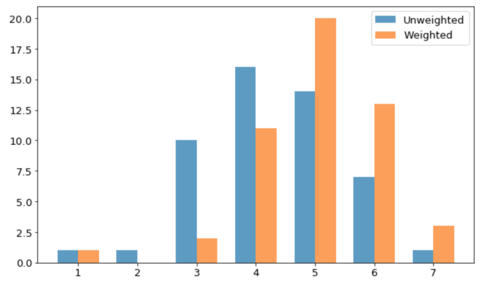


Figure 3: Preference for sessions with more domains

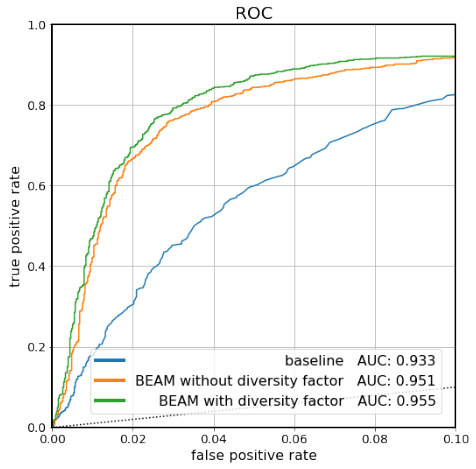


Figure 4: Effect of diversity factor

## References

[1] G. Bassett, C. D. Hylender, P. Langlois, A. Pinto, and S. Widup, "2020 verizon data breach investigations report," 2020.

[2] E. Ahlm and A. Litan, "Market trends: User and entity behavior analytics expand their market reach," 2016.

[3] Ibm qradar user behavior analytics (uba). [Online]. Available: https://www.ibm.com/security/security-intelligence/qradar

[4] Arcsight user behavior analytics. [Online]. Available: https://www.microfocus.com/en-us/products/security-operations/overview

[5] Splunk insider threat detection tools. [Online]. Available: https://www.splunk.com/en_us/cyber-security/insider-threat.html

[6] H. Eldardiry, E. Bart, J. Liu, J. Hanley, B. Price, and O. Brdiczka, "Multi-domain information fusion for insider threat detection," in *Security and Privacy Workshops (SPW), 2013 IEEE*. IEEE, 2013, pp. 45–51.

[7] B. Böse, B. Avasarala, S. Tirthapura, Y.-Y. Chung, and D. Steiner, "Detecting insider threats using radish: A system for real-time anomaly detection in heterogeneous data streams," *IEEE Systems Journal*, vol. 11, no. 2, pp. 471–482, 2017.

[8] D. Haidar and M. M. Gaber, "Data stream clustering for real-time anomaly detection: An application to insider threats," in *Clustering Methods for Big Data Analytics*. Springer, 2019, pp. 115–144.

[9] A. Zargar, A. Nowroozi, and R. Jalili, "Xaba: A zero-knowledge anomaly-based behavioral analysis method to detect insider threats," in *Information Security and Cryptology (ISCISC), 2016 13th International Iranian Society of Cryptology Conference on*. IEEE, 2016, pp. 26–31.

[10] V. Kalyan, A. Ignacio, C. Alfredo, K. Vamsi, B. Costas, and L. Ke, "Ai2: Training a big data machine to defend," in *IEEE International Conference on Big Data Security, New York, NY, USA*, 2016.

[11] A. Tuor, S. Kaplan, B. Hutchinson, N. Nichols, and S. Robinson, "Deep learning for unsupervised insider threat detection in structured cybersecurity data streams," *arXiv preprint arXiv:1710.00811*, 2017.

[12] L. Liu, O. De Vel, C. Chen, J. Zhang, and Y. Xiang, "Anomaly-based insider threat detection using deep autoencoders," in *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*. IEEE, 2018, pp. 39–48.

[13] T.-F. Yen, A. Oprea, K. Onarlioglu, T. Leetham, W. Robertson, A. Juels, and E. Kirda, "Beehive: Large-scale log analysis for detecting suspicious activity in enterprise networks," in *Proceedings of the 29th Annual Computer Security Applications Conference*. ACM, 2013, pp. 199–208.

[14] M.-L. Shyu, S.-C. Chen, K. Sarinnapakorn, and L. Chang, "A novel anomaly detection scheme based on principal component classifier," Maimi Univ Coral Gables Fl Dept Of Electrical and Computer Engineering, Tech. Rep., 2003.

[15] M. A. Maloof and G. D. Stephens, "Elicit: A system for detecting insiders who violate need-to-know," in *International Workshop on Recent Advances in Intrusion Detection*. Springer, 2007, pp. 146–166.

[16] I. A. Gheyas and A. E. Abdallah, "Detection and prediction of insider threats to cyber security: a systematic literature review and meta-analysis," *Big Data*

*Analytics*, vol. 1, no. 1, p. 6, 2016.

[17] R. F. Smith, “Randy franklin smith’s ultimate windows security,” 2011.

[18] J. J. Eberhardt, “Bayesian spam detection,” *Scholarly Horizons: University of Minnesota, Morris Undergraduate Journal*, vol. 2, no. 1, p. 2, 2015.

[19] D. Zhou, J. He, K. S. Candan, and H. Davulcu, “Muvir: Multi-view rare category detection.” in *IJCAI*, 2015, pp. 4098–4104.