

Detecting Malicious Insiders through ML-based Behavior Analytics

Sanket Nayak, Department of CSE, GIET University, Gunupur, Odisha, India, sanketnayak@giet.edu
Dr. Dhikhi T, L&T EduTech, Chennai, India, dhikhi.t1@lntec.com
Smruti Suman Badjena, Department of CSE, GIET University, Gunupur, Odisha, India,
smrutisumanbadjena@giet.edu

Abstract: Modern organizations have to face growing risks posed by insider threats, where the person with valid access abuses it either intentionally or unintentionally. The rule-based and signature-based methods of securing a system have shown their limited ability to detect those behaviors since they tend to look like usual actions. In this paper, a behavior-based approach to detecting insider threat using User Behavior Analytics (UBA) with unsupervised machine learning is presented. Using the CERT Insider Threat Dataset (R4.2), logs about users' logon events, file accesses, emails, and web accesses are pre-processed, merged, and converted into behavioural and aggregated user features. Local Outlier Factor (LOF) and One-Class Support Vector Machine (OC-SVM) are used to describe normal behavior and detect anomalies; the results are visualized with the help of ELK Stack (Elasticsearch, Logstash, Kibana). The experiment shows that LOF can successfully catch sudden bursts in behavior, such as unusual off-hours logins or abnormal amounts of file accesses, while OC-SVM is more sensitive to behavioural changes.

Keywords—Insider Threats; User Behaviour Analytics; Machine Learning; Anomaly Detection; CERT Dataset; ELK Stack; Cybersecurity.

1. Introduction

Networks of enterprise-level companies and cloud infrastructure are generating more and more data, but growth does not make the security picture easier. The vast majority of security professionals still think about security in terms of perimeter—if we do not let the attacker in, the internal space is safe. This breaks once the attacker holds a valid credential. Increasing number of security breaches happen from people who must not be restricted in their access from outside—the employees, contractors, or users whose accounts have been stealthily compromised [1-3]. The regular firewalls and traditional SIEMs based on signatures did not anticipate such threat—they are looking for known patterns of a bad behavior, not for good-user who does something strange.

The User Behavior Analytics (UBA) takes another approach. Instead of asking whether the action corresponds to a known attack pattern, it asks whether the action conforms to usual behavior of the particular user. Every user has his/her behavioural baseline, and the system watches for the deviations from it. And once used together with unsupervised learning, it starts being practically useful, because this technique allows detecting insider's activity which is never seen before and does not need labelled data of past attacks—which are rare enough. This is the basic principle of this paper. We propose a pipeline that collects logs from various sources, extracts behavioural features from them, performs analysis using two unsupervised detectors—LOF and OC-SVM—in parallel, and outputs the results into the ELK stack so the suspicious events become understandable for human analyst.

In Section II we discuss the related works; in Section III we describe the methodology and architecture of our system; the results are shown in Section IV, and some conclusions and directions for future work are provided in Section V.

2. Related Work

Rule-based SIEMs and static access thresholds were the first real line of defence against insider misuse, and in a narrow sense they still earn their place — known bad patterns get caught reliably. The trouble is everything else: rules have to be written and maintained by hand, they miss anything that develops slowly, and they generate enough false positives to wear an analyst down [4]. UBA systems moved the needle by building a behavioural baseline per user instead of relying on a fixed rulebook, drawing on authentication, file, email and web logs to score risk in context [5], [6].

A few groups have pushed this further, pairing behavioural baselining with a boundary-based One-Class SVM detector built specifically for enterprise insider attacks [7], and more recent work has started extending that baselining toward near-real-time prediction of insider risk [8]. Even so, a lot of this work sits behind closed doors — vendor systems with no published evaluation — and the results that do exist tend to depend heavily on how well the underlying features were engineered in the first place. On the modelling side, density-based detectors such as Local Outlier Factor [9] catch anomalies by checking whether a point sits in a noticeably sparser neighbourhood than its peers, which works well for subtle, localized weirdness. One-Class SVM [10] takes a different angle: it learns the boundary of what “normal” looks like and flags anything outside it, which tends to do better on gradual drift than on one-off spikes.

Isolation Forest offer a third option, isolating rare points through recursive partitioning of the feature space [2], and a handful of studies have run classical unsupervised pipelines like these directly against the CERT dataset [8]. Deep architectures — autoencoders [3], [9], LSTM/GRU sequence models [1], transformer-based log-correlation approaches [4], [10] pick up richer temporal structure but need more labelled data and compute than most SOC deployments can spare. Several studies also wire detection output directly into Elasticsearch and Kibana so results are usable on screen rather than buried in a CSV [11-13].

3. Proposed Methodology

Dataset: In order to test these techniques, we take the CERT R4.2, an insider-threat dataset of a whole organization (logons, file access, e-mail, browsing) compiled by Software Engineering Institute of CMU [11]. It is useful in spite of fact that it's synthetic to report. As of today, there aren't a many similar datasets so CERT R4.2 got almost to a standard reference in such works.

Pre-processing and Feature Engineering: The raw logs needed a lot of cleaning before anything of value could be obtained. We dropped null or corrupt records, filtered out system-level noise irrelevant to human activity, normalized all timestamp fields to hour/day/week for cross-log comparisons, and unified the categorization fields (e.g., department, file type) for each attribute across different logs. Then, all the logs were merged into a user-centric session-level table indexed by ID and time, which led to two categories of features: behavioural features describing logins, file access habits, and time spent online/in emails; aggregated features like time-of-day counts, time normalization by role/peer-group, and daily/weekly summaries.

Anomaly Detection Models: We then used two unsupervised, non-linear models. We ran each trained to learn behaviour only during sessions considered "normal" over our whole log file:

LOF (Local Outlier Factor) is to compare each session's density with neighbouring sessions, thus identifying behaviours with markedly low density relative to their surroundings. This identifies activity spikes such as increased file activity or login attempts outside of normal working hours, and even a wave of a typical web access or email behavior and One-Class SVM (OC-SVM) on the other hand learns the boundary of what normal behaviour looks like and anything outside of it is flagged. It tends to pick up slower changes, for example a steady increase in web activity over several days - that LOF, with its focus on local density, can miss.

System Architecture and ELK Visualization: There are nine steps to the pipeline: get and clean the data; prepare the space; data transformation and profiling; behaviour feature extraction; run LOF and OC-SVM; push detected anomaly scores to ELK Stack; see how it looks on the other end.

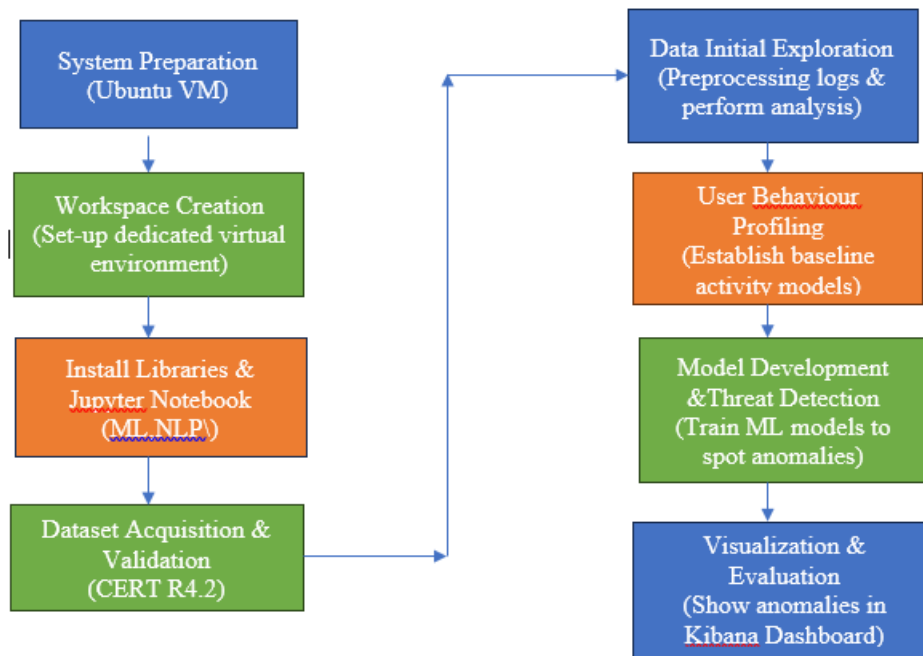


Figure 1: End-to-end system architecture of the proposed framework.

Elasticsearch has all the feature engineering and anomalous behavioural data; Logstash processes the cleaned CSV/JSONs for consumption by ES and other ELK stakeholders; then Kibana gives the user the power to login-timelines, file access behaviours, an anomaly score indicator, and fine-grained filtering per user so that an analyst can drill down and see why a particular login session triggered. The whole pipeline shown in Fig. 1 operates as a closed loop – behavioural changes that trigger a particular run of LOF/OC-SVM also influence future model training; detection can trigger visualization and threat analysis which can inform more appropriate behavioural features or prompt another round of data collection/feature selection.

It is to implement a closed-loop architecture, in order to supply the data validation and user-behaviour profiling processes with ongoing logs from future Data Initial Exploration stages; consequently, the base behavioural parameters used are not static, but will continue to evolve as new logs come in. Such a structure facilitates extensions into near real-time prediction and

monitoring in Section V, as it is only required to schedule future invocations of the same feedback loop rather than redesigning the whole process.

4. Results and Discussion

The two models were then trained on non-anomaly behavior sessions and tested on the merged data from the entire dataset. Since no labeled data exists for this experimental configuration for CERT R4.2, performance is evaluated using anomalousness-score distributions, number of flagged sessions, quality of separation between clusters, and manual behavioral verification, rather than using supervised techniques like precision and recall. LOF did frequently find intense, distinct anomalies; one such case is a single user who opened 47 files within a day, compared with that user's baseline of 5–8/day; this is a straightforward, hard alert. LOF found a login at 2:13 AM for another user, which is obviously outside their typical work hours. OC-SVM found a slow drift; another set of sessions consisted of a four-day cumulative upward trend in web access, coupled with a slow rise in external email, but where the absolute count for either never got extreme. The comparison is captured more precisely by Table I and visualized with plots (each session as a point) for LOF versus OC-SVM score in Fig. 2.

TABLE I. Comparison of LOF and OC-SVM on behavioral anomaly detection

Aspect	LOF	OC-SVM
Detection basis	Local density deviation	Distance from learned normal boundary
Sensitive to	Sharp, isolated spikes	Gradual behavioural drift
Anomaly profile	Diverse, high-impact outliers	Compact, consistent outlier set
Limitation	Sensitive to neighbourhood size	Sensitive to kernel/hyper-parameters

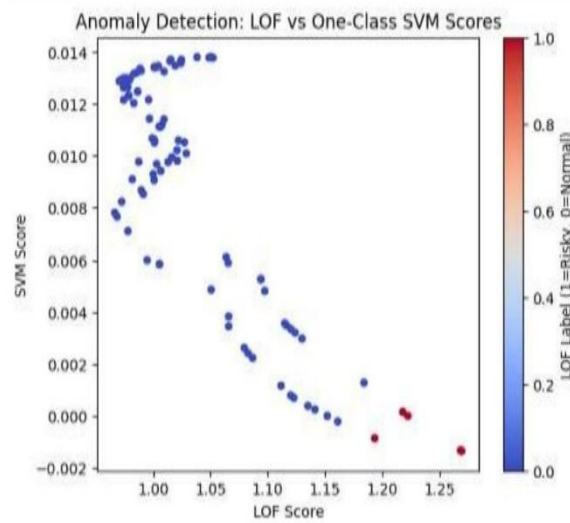


Fig. 2. LOF score vs. OC-SVM score per session, coloured by LOF risk label

As Fig. 2 shows, the two scores are inversely correlated but not perfectly, so most sessions cluster in the low-LOF/high-SVM-score region (normal behaviour), while the small set of high-LOF, near-zero-SVM-score points in the lower-right corner corresponds to the sessions that both models agree should be marked as anomalous. Even though LOF and OC-SVM calculate risk using fundamentally different methods (the first is a local-density-based, the second is a boundary-based), the fact that most of high-risk sessions created by both methods can be observed with only handful of red-label points getting outside the main blue cluster confirms that indeed these two notions "isolated" for LOF and "outside normal boundary" for OC-SVM coincide on this small set.

If you look at the results as a whole, the results indicate that LOF and OC-SVM are not competitors that addresses different behaviour trigger profiles per definition: LOF catches extreme, fast developing incidents well (i.e. This "combine both" approach -- as opposed to using only one model -- expands detection coverage without the resource penalty of deep sequence models, and the ELK integration means that this expanded coverage is presented as a single coherent dashboard to analysts rather than two disparate outputs.

5. Conclusion

An insider threat detection framework leveraging User Behaviour Analytics, unsupervised Machine Learning using LOF and OC-SVM, and ELK-based visualization was presented in this paper. What You Will Learn How to process raw, multi-source CERT R4. 2 Logs into structured behavioural features A framework for detecting meaningful anomalies without any labelled attack data, leveraging Kibana dashboards that can be understood by Analysts. The results indicated that LOF detects sharp spikes in behavior, whereas OC-SVM finds gradual drift and hence the two models are complementary. The main limitation of the framework is that it was based on a synthetically generated data set and offline analysis whereas any deployment in an organization would be validated against live enterprise logs and require real-time ingestion. With your future work looking at sequence models (LSTM/Autoencoder) for temporal patterns, role-based and HR-contextual features, ELK alerting in real-time, and a risk-scoring capability to help SOCs prioritising alerts as part of the journey from this framework towards operational deployment in enterprise Security Operations Centres.

References

1. R. Li, M. Zhang, and F. Yu, "An LSTM-driven framework for sequential insider threat detection using enterprise activity logs," *Computers & Security*, vol. 109, p. 102385, 2021.
2. S. E. K. Smith and J. Brown, "Deep autoencoder-based anomaly detection for insider threats using user behaviour logs," *IEEE Access*, vol. 8, pp. 210450–210462, 2020.
3. W. Iqbal, R. Latif, and M. Afzal, "Behaviour-based insider threat detection using deep learning: Challenges and opportunities," *IEEE Access*, vol. 12, pp. 55360–55378, 2024.
4. H. M. Islam and N. Akhter, "A hybrid UBA and one-class SVM approach for insider attack detection in enterprise environments," *IEEE Trans. Information Forensics and Security*, vol. 17, pp. 1224–1236, 2022.
5. M. Breunig, H.-P. Kriegel, R. Ng, and J. Sander, "LOF: Identifying density-based local outliers," in *Proc. ACM SIGMOD*, 2000, pp. 93–104.



6. B. Schölkopf, J. Platt, J. Shawe-Taylor, A. Smola, and R. Williamson, “Estimating the support of a high-dimensional distribution,” *Neural Computation*, vol. 13, no. 7, pp. 1443–1471, 2001.
7. P. Roy, S. Chattopadhyay, and K. Banerjee, “Unsupervised machine learning techniques for identifying malicious insiders using the CERT dataset,” *Expert Systems with Applications*, vol. 188, p. 116012, 2022.
8. M. Theis, “The CERT Insider Threat Datasets: Design, Challenges, and Opportunities,” *Carnegie Mellon University*, 2019.
9. R. Banerjee et al., “Log analysis and anomaly detection in enterprise networks using Elasticsearch and Kibana,” *Procedia Computer Science*, vol. 218, pp. 1452–1460, 2023.
10. S. Ghorbani and A. Fathi, “SIEM-based anomaly detection using ELK stack and machine learning,” *Int. J. Information Security Science*, vol. 11, no. 2, pp. 65–78, 2022.
11. A. Rashid, J. Chivers, and R. Cook, “A UBA-driven insider threat detection architecture for enterprise networks,” *Computers & Security*, vol. 112, p. 102526, 2022.
12. S. Song and E. Stolfo, “Understanding user behaviour patterns for anomaly-based insider threat detection,” in *IEEE Security & Privacy Workshops*, 2020, pp. 302–309.
13. A. Alam, S. A. Butt, and T. Hussain, “A comprehensive survey on insider threat detection: Techniques, challenges, and research directions,” *J. Network and Computer Applications*, vol. 240, p. 103104, 2025.