International Journal of Computer Techniques–IJCT Volume 13 Issue 2, March-April - 2026

Open Access and Peer Review Journal ISSN 2394-2231 https://ijctjournal.org/

Integral Techniques to Analyze Security Data and Predict Cyber Attacks

Manikandan Sampathkumar
Xforia Inc, Dallas, Texas
Email: mani.s@xforia.com

----------**********----------

Abstract:

This study explores integral techniques for analyzing security data and predicting cyber attacks, emphasizing the convergence of statistical analysis, machine learning, data mining, and threat intelligence to enhance cyber defense capabilities. Integral analytical techniques aim to unify multiple data analysis methodologies into a coherent framework that can extract meaningful patterns, detect anomalies, and infer malicious intent. By integrating data preprocessing, feature extraction and advanced analytical models, organizations can transform raw security data into actionable insights. Statistical and probabilistic techniques form the foundation of security data analysis by enabling baseline modeling of normal system behavior and identification of deviations that may indicate malicious activity. Methods such as Mahalanobis distance, Markov models, entropy-based measures, and time-series analysis are widely used to detect anomalies and assess risk. These techniques provide interpretability and mathematical rigor, allowing security analysts to quantify uncertainty and evaluate the likelihood of potential threats. A key aspect of integral security analysis is the correlation of heterogeneous data sources. Individual security events often appear benign in isolation but reveal malicious intent when correlated across multiple dimensions. This study contributes to the growing body of research that advocates for intelligent, proactive, and holistic cybersecurity solutions capable of anticipating and mitigating future cyber threats.

Keywords — **Cyber Threat, Predictive Analysis, Threat Intelligence Integration, Statistical Models, Machine Learning, Security Behavior Analytics**

----------**********----------

I. INTRODUCTION

Even though strict company security policies and education can stop cyberattacks to some extent, an unorganized cyber environment stands little to no chance against malwares, insider threats or zero-day exploits. In order for an organization defend against such threats effectively and stop cyber-attacks before they manifest themselves, they need to have integral techniques capable of analyzing and predicting security data within a critical timeframe. This poses a lot of challenges like analyzing vast amount of security data, unify them under a single pane of glass, detecting anomalies, integrating user behavioral features with host and network data, correlation of asset information and finally an affordable big data ecosystem capable of handling petabytes of data while remaining within established security budget constraints. With an abundance of threats hanging loose, only 14% of companies are capable of mitigating cyber-attacks [1]. We are clearly in need of intelligent cyber security solutions that can combat external and internal threats faced by organizations. In this paper, I will outline the architecture, underlying mathematical algorithms, and open-source tools that organizations can leverage to develop their own

intelligent security framework—one that can ingest, analyze, and correlate data from multiple sources and intelligently aggregate insights across the entire information set.

II. BACKGROUND AND ARCHITECTURES

Madhu Shashanka and Charles Swab et al [2] use SVD which stands for Singular Value Decomposition to analyze user data. SVD's are machine learning algorithms that can detect automatically detect anomalous behavior. Madhu Shashanka and Charles Swab et al [2] came up with the following architecture for user behavioral analytics or UBA for short.

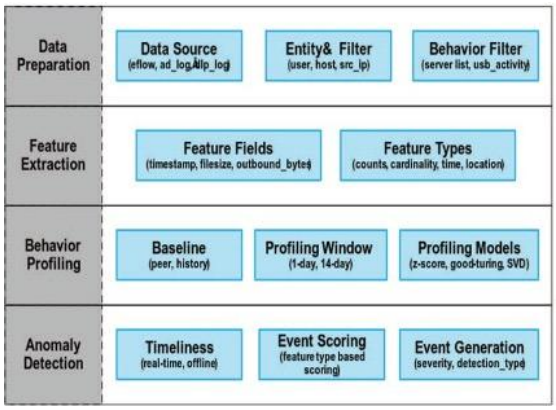

Fig 1: Architecture of UBA [2]

The proposed architecture is composed of four distinct stages, beginning with data preparation. A critical component of the research is the selection and integration of data sources, as both the volume and quality of data directly influence the effectiveness of anomaly detection. High-quality datasets enable a deeper understanding of user behavior, which in turn enhances the accuracy of behavioral modeling and predictive analysis. This perspective aligns with the findings of Aninitida Khade [3], who emphasizes the necessity of Big Data for effective customer behavior analytics. The next stage involves data extraction, in which the collected information is decomposed into key attributes such as timestamps, user identifiers, location, data transfer volume, downloaded content, and related fields. User identifiers are then mapped to corresponding usernames using maintained employee records. At this point, user data from multiple domains across the enterprise is normalized and prepared for analysis. Before performing analytics, a behavioral profile is established by correlating user data across different activities. This correlation is conducted over a configurable profiling window and can be adjusted to align with organizational requirements. After this step, behavioral modeling begins, incorporating techniques such as SVD-based behavior profiling models. In the final phase, correlation values are generated and scored against each individual's established behavioral profile. The greater the deviation or anomaly exhibited by a user, the higher the resulting score. Machine learning techniques can further enhance the system by improving its ability to learn and interpret behavioral patterns. As also noted by Glenn M. Lambert II [4], deep learning approaches can be applied to detect and prevent cyberattacks; these methods focus on learning data representations rather than task-specific rules and form part of the broader family of machine learning techniques. One platform currently used by security analysts, as described by Madhu Shashanka et al. [2], is the Niara Security Analytics platform. It supports threat hunting, incident investigation, and the identification of behavior-based anomalies. The primary objective of the Niara platform is to deliver automated detection of cyberattacks, which is essential because such attacks often bypass perimeter defenses and originate from within the organization.

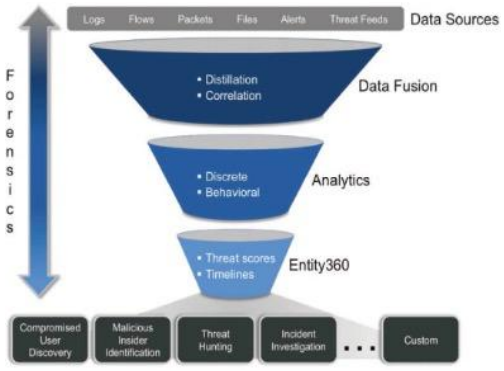

Fig 2: Overview of the Niara architecture [2]

Shashanka et al. [2] describe the Niara architecture for cyber forensics using a top-down

approach, as illustrated in Figure 6. In this model, data from multiple sources is collected and fused, encompassing raw inputs as well as a combination of network and security data. The fusion stage involves correlating raw data to enhance its relevance and interpretability. This is followed by distillation, where the data is summarized to provide enriched context, further augmented through meaningful interpretation of metadata. As a result, the original raw data is transformed into an enriched dataset that enables analysts to derive clearer and more actionable insights. Additionally, Anindita A. Khade [3] and Glenn M. Lambert II [4] emphasize the importance of incorporating a preprocessing stage for certain data sources. This preprocessing ensures uniform data formatting across all inputs entering the system, allowing analysts to view and analyze diverse data sources in a consistent manner. Such standardization simplifies correlation and analysis, regardless of the tools or log sources from which the data originates.As described by Shashanka et al. [2], the next stage of the Niara architecture is the analytics layer. This layer is broadly divided into behavioral analytics and discrete analytics. Discrete analytics modules typically rely on supervised machine learning techniques and are primarily used to detect and classify known threats, such as malware infections and suspicious executables. In contrast, behavioral analytics modules largely employ unsupervised machine learning methods and are designed to tackle more complex challenges, particularly the identification of unknown threats, including malicious or compromised insiders. Once the system is fully configured, it generates events and alerts whenever malicious activity or anomalous behavior is detected. These events are linked to one or more entities and are accompanied by two metrics: a severity score and a confidence score [2]. The severity score is defined by the analyst to reflect the potential business impact on the organization, while the confidence score represents the system's assessment of the likelihood that the detection is accurate. To enable near real-time alerting, the system must rapidly ingest data from multiple integrated log sources and process it through the Niara architecture to analyze and flag suspicious activity. This necessitates a high-performance system capable of capturing large volumes of data from numerous sources and executing analytics at scale. Hsinchun Chen et al. [6] and Anindita A. Khade [3] propose Apache Hadoop as an effective solution. Apache Hadoop is an open-source platform with a robust ecosystem of tools specifically designed for large- scale data storage, processing, and analysis. Several Hadoop components are particularly well suited for large-scale data storage, processing, and analytics, including HDFS (Hadoop Distributed File System), Kafka, MapReduce, Spark, HBase, and Oozie, among others. From this set, a select subset will be examined in greater detail to demonstrate how they can be leveraged effectively within a UBA architecture. Anindita A. Khade [3] emphasizes the importance of HDFS and MapReduce within big data architectures. HDFS is used for long-term storage of large datasets, leveraging a distributed and easily scalable design. Khade [3] proposes an architecture in which data initially flows into HDFS and is then processed through MapReduce. MapReduce is a programming model that splits input data using a Mapper function and subsequently aggregates the results using a Reducer function. This approach enables faster data processing by distributing workloads across multiple nodes and executing tasks in parallel. After the MapReduce phase, the processed data is forwarded for rule generation and analytical processing, with results presented through a web-based user interface supported by visualizations such as charts and graphs. In contrast, Abdul Rauf Baiga et al. [7] advocate the use of Apache Spark for more efficient parallel processing compared to MapReduce. While Spark is based on similar distributed computing principles, it differs in execution by operating primarily in memory, whereas MapReduce relies on disk-based processing. The disk I/O overhead associated with MapReduce results in slower performance, particularly due to frequent read and write operations during the mapping and reducing phases. As a result, Spark is estimated to be up to 100 times faster than traditional MapReduce implementations [8]. Other notable tools include Kafka and Oozie.

Kafka serves as a messaging bus, acting as a data pipeline until the information is consumed by downstream processes. It is highly scalable, fault-tolerant, and fast, making it ideal as an intermediary layer for moving data between storage systems and analytical tools while effectively managing latency. Oozie, on the other hand, is a workflow scheduler that orchestrates Hadoop jobs according to predefined schedules. Behavioral analytics tasks and other algorithms can be integrated into Oozie to run automatically at specified times. Figure 7 presents an overview of the Hadoop ecosystem [9], illustrating the various tools designed to address specific areas within the big data domain. For storage, tools such as HDFS, Hive, and HBase are commonly used, while data management is handled by components like Zookeeper, Avro, Oozie, and Whirr.

Fig 3: Overview of big data ecosystem with Hadoop(13)

Big data processing relies on frameworks such as MapReduce and Spark, and distribution is managed by platforms including Cloudera, Hortonworks, MapR, and IBM BigInsights. For deriving insights, tools like Mahout, Hue, and Beeswax are utilized, whereas data integration is facilitated by Sqoop, Flume, Hiko, and Chukwa. Finally, programming and querying within the Hadoop ecosystem are supported by languages and frameworks such as PIG, Hive QL, and Jaql. These are widely available open-source tools that can be adopted and utilized by any organization. Together, these tools provide a comprehensive ecosystem for handling the diverse requirements of big data applications.

III. BEHAVIOR DETECTION METHODS

A. Mahalanobis distance

Let us assume the variable used to calculate the behavioral scores follows a normal distribution. If this is the case, low probability events (like an anomaly) would occur in the tail of the distribution far away from the mean. The farther an event is away from the mean, the grater is the magnitude of the outlier. When this distance is represented in terms of standard deviation rather than units, we get the z-score. Z-score tells us how many standard deviations away an outlier is from the mean value. This makes comparison between anomalies in variables with different distributions. Mahalanobis distance is a grandiose version of the z-score. When multiple variables are involved in each of the observation, Mahalanobis distance helps us to find how many standard deviations away a certain observation is present from the mean value of all the other observations. Let us take N variables, represented by x = {x1, . . . ,xN} and a few observations to go with variables, represented by X = {x1, . . . , xK} and also a mean vector, represented by μ = {μ1, . . . , μN} followed by the covariance matrix Σ. In this scenario the Mahalanobis distance is given by the equation:

$$\sqrt{(x-\mu)^T \Sigma^{-1} (x-\mu)} \qquad (1) \qquad [2]$$

TABLE I. ALGORITHMS USED

Step 1.	Inputs: Baseline matrix $\mathbf{X}$, test vector $\mathbf{x}$
Step 2.	z-score $\mathbf{X}$ and preprocess $\mathbf{x}$: $\bar{\mathbf{X}} \leftarrow zsc(\mathbf{X}, \mathbf{X})$, $\bar{\mathbf{x}} \leftarrow zsc(\mathbf{x}, \mathbf{X})$
Step 3.	SVD: $\bar{\mathbf{X}} \rightarrow \mathbf{USV}^T$
Step 4.	Number of components: compute r from Eqn. 5 or 6
Step 5.	Reduce to first r components: $\mathbf{U} \rightarrow \bar{\mathbf{U}}$, $\mathbf{S} \rightarrow \bar{\mathbf{S}}$
Step 6.	Project to SVD space: $\bar{\mathbf{U}}^T \bar{\mathbf{x}} \rightarrow \mathbf{y}$
Step 7.	Compute distance: $\mathbf{y}^T \bar{\mathbf{S}}^{-2} \mathbf{y}$.

The flowchart of the algorithms used in order is show in table1 [2]. The sequential step approach mentioned above are made a little bit more flexible. This is done so that they become appropriate for the real-world use-cases. For this purpose, Mahalanobis formula is enhanced [11]. Some of the real word use cases include one side deviation, variable weighting, robustness to outliers etc. We will be looking at the real-world use cases and the formulas used on them a little bit more in detail.

B. Markov Models

Markov models represent the likelihood of transitioning from one stage of kill chain phase to the subsequent phase. The different stages of the security outline which is also referred to as cyber kill chain are reconnaissance, weaponization, delivery, exploitation, execution and command and control.

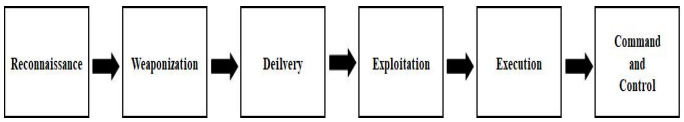

Fig .4 Cyber kill chain

The cyber kill chain gives us an outline of how various stages of threats and malicious activities can be classified. One of the most prominent models that discusses Markov model is a state based stochastic model described by Abraham, Subil, and Suku Nair [5]. In fig 17, Abraham et al [5] describes analytics model for cybersecurity for state based stochastic modeling [5]. The Attack Graph is constructed using a network model builder that incorporates network topology, the services deployed on each host, and a set of attack rules derived from vulnerabilities associated with those services. Each edge in the Attack Graph is assigned a probability representing the likelihood that an attacker will successfully exploit a given vulnerability. A stochastic process is then applied to the Attack Graph to model potential attack behaviors. This approach enables the extraction of multiple security metrics, providing valuable insights into the current security posture of the network. Additionally, the model accounts for zero day attacks by considering unknown or previously undisclosed vulnerabilities.

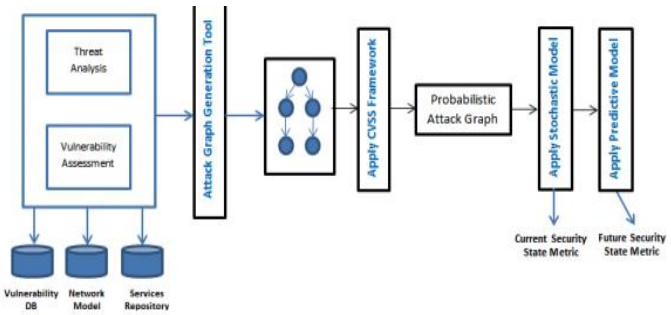

Fig .5 Cyber analytics model [5]

C. Data Flows and Predictive Analysis

Fig 6 represents the data flowchart for the behavior analytics process. As mentioned in the figure, the process takes effect from collecting log sources from different applications the user is using. This step 1, usually has log sources ranging from the user's end point to their activity over the network. This tells us exactly how the users behave with the applications within and how they use the network and their activity outside the organization perimeters (network boundary). Step 2 would be to parse and normalize the incoming log sources. Since these log files would be of a different format from each other, we need to parse and normalize them in order to get them to a single format. After parsing, usually the attributes are mapped to certain

fields that are used in ad hoc searches/ policies later on. These fields are custom built into a unified system used as a front-end application to this process. Step 3 deals with storing the parsed and normalized data. This storing can be done in two aspects. Since we will be mostly dealing with real time analytics, a storage of a period of 3 months could be used for fast and efficient analysis. We will discuss about this process later in detail in the next chapter as to how this is helpful for ad-hoc searches and policies running on the background. For a period of more than 3 months, the data can be moved to a database for robust long-term storage. Long term storage would be more archival focused rather than suiting real time analytics. Step 4 and step 5 of the flowchart relate to the details mentioned in step 3. In step 4 the data elements and attributes of the user may be analyzed to create a Resource Description Framework (RDF). Usually the analytics is done through policies that are run through the incoming data. In Step 5 the data is stored in graph database, graph database can be helpful for visualization purposes.

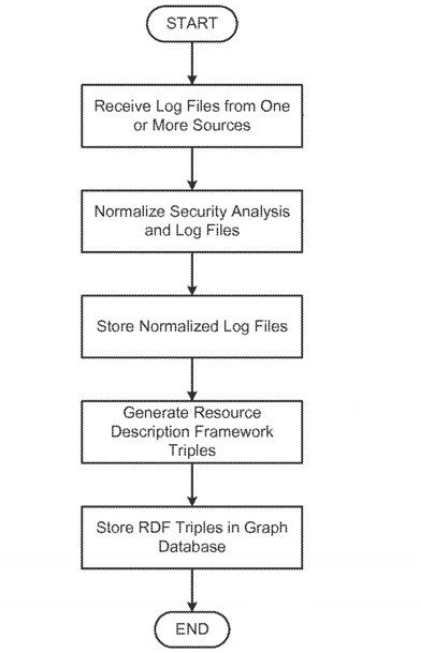

Fig. 6 Behavioral Analytics Flow Chart [10]

In this paper we have come across the various architecture diagrams for UBA including the Niara architecture which give an outline as to how to detect insider threats from the data obtained from various log sources. We also saw how Hadoop as an open source platform, had the required tools to integrate, analyze, process and store big data. In this section we are going to look at the algorithms and methodology that runs in the analytics side of things. As mentioned earlier by Shashanka et al [2] the essential data from the log sources required are timestamp of first access of the day, timestamp of last access for the day, duration between last and first access, sum total of durations of all eflows of the day, number of eflows during the day, total upload bytes and total download bytes. The data from the above sources serve as an input to the anomaly detection algorithm. There are two types of baselines formed using this algorithm which are the historical baseline and the peer baseline respectively. For historical baseline, individual user-server pair's data for several days are used as a baseline data X. For peer baseline, data vectors for all users accessing the server on a certain day is used as baseline X for the day.

D. Identity analytics in sorting user behavior

Analytics on an individual's identity can also be used to sort user behavior. Figure 11 shows a flow diagram, it starts from requesting identity data when a subscriber requests access to an identity service provider. The process then continues to show how the identity directories are stored to how policies and analytical threat models are run on individual's patterns.

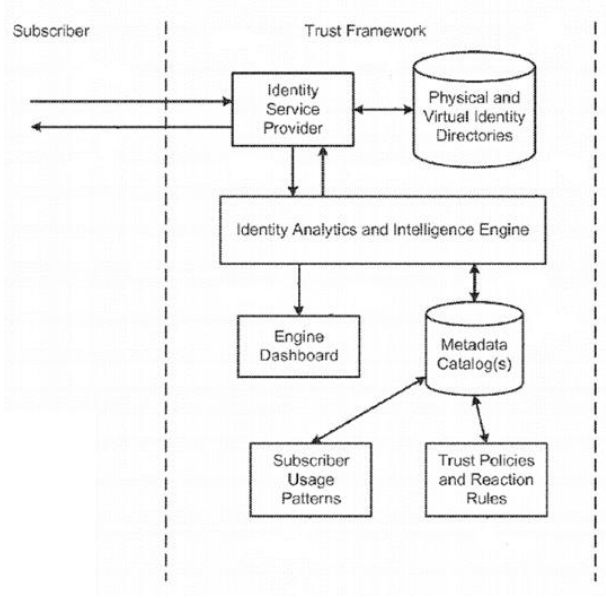

Fig 7. Process data flows associated with a subscriber [10]

In many cases identity data can also be coupled with entitlement data so that we are aware of the individual's position in the organization. When an individual's position in an organization is known, they can be used to run peer group analysis to determine how the behavior of the individual is different from other peers with the same access level of the individual. There is a behavioral profile that is formed and when there is a deviation, we know that the individual's behavior is way off from other peers with similar access level. This can later be investigated and resolved.

IV. CONCLUSION

This paper highlights the need for cost efficient, intelligent, scalable, and predictive cybersecurity solutions capable of operating within complex and data-intensive enterprise environments. Effective cyber defense now requires the seamless integration of heterogeneous security data, advanced mathematical and statistical models, and behavioral analytics to move from reactive incident response to proactive threat anticipation. In response to this need, this paper presents a comprehensive intelligent security framework that combines robust data ingestion pipelines, correlation mechanisms, and analytical algorithms with a cost-efficient open-source ecosystem. By detailing the system architecture, underlying mathematical foundations, and practical implementation considerations, this work demonstrates how organizations can achieve enhanced visibility, timely detection of anomalous activities, and informed decision-making, thereby significantly improving their ability to defend against both external attacks and insider-driven threats.

REFERENCES

[1] **Mansfield, Matt.** Cyber Security Statistics: Numbers Small Businesses Need to Know. *Small Business Trends.* [Online] January 24, 2019. https://smallbiztrends.com/2017/01/cyber-security-statistics-small-business.html.

[2] *User and entity behavior analytics for enterprise security.* **Madhu Shashanka, Charles Schwab, Min-Yi Shen, Jisheng Wang.** Washington DC, USA : IEEE, 2016. 978-1-4673-9006-4.

[3] *Performing Customer Behavior Analysis using Big Data Analytics.* **Khade, Anindita.** s.l. : Procedia Computer Science. 79. 986-992. , 2016. 10.1016/j.procs.2016.03.125.

[4] *Security Analytics: Using Deep Learning to Detect Cyber Attacks.* **II, Glenn M. Lambert.** Master's thesis, s.l. : Digital Commons, 2017. 2572-5874.

[5] Abraham, Subil, and Suku Nair. "Cyber security analytics: a stochastic model for security quantification using absorbing markov chains." *Journal of Communications* 9.12 (2014): 899-907.

[6] *Business intelligence and analytics: from big data to big impact.* **Hsinchun Chen, Roger H. L. Chiang, Veda C. Storey.** 4, Minneapolis, USA : MIS Quarterly, 2012, Vol. 36 .

[7] *Big data analytics for behavior monitoring of students.* **Abdul Rauf Baiga, Hajira Jabeenb.** Riyadh : ScienceDirect, 2016, Vols. Procedia Computer Science 82 (2016) 43 – 48 .

[8] **Neumann, Saggi.** Spark vs Hadoop Mapreduce. *XPlenty.* [Online] November 24, 2014. https://www.xplenty.com/blog/apache-spark-vs-hadoop-mapreduce/.

[9] **Barrett, Gregg.** Building a Big Data platform with the Hadoop ecosystem. *slideshare.* [Online] July 21, 2015. https://www.slideshare.net/sirghbarrett/building-a-big-data-platform-with-the-hadoop-ecosystem.

[10] **Paul Dennis Bailor, Eric Louis Uythoven.***CHARACTERIZING USER BEHAVIOR VIA INTELLIGENT DENTITY ANALYTIC. US 9,679,125 B2* Colorado, USA, June 13, 2017.

[11] **Ker, Andrew D.** STABILITY OF THE MAHALANOBIS DISTANCE: A TECHNICAL NOTE. 2010, Computing Science Group, Oxford University.