User behavioral analysis using machine learning algorithms

Artem Sharkota, majoring in Cyber security
College of Applied Computing
Michigan Technological University

Introduction

My project is centered around user behavior analysis, an essential aspect of modern cybersecurity. With various techniques, understandings, and approaches, we can identify and recognize anomalies in user behavior in cases of data or credential breaches, fraud attempts, and other potential threats to a company. By evaluating user behavior, we can not only improve the user experience but also gain valuable insights into how users interact with systems, applications, or websites, helping to identify abnormal behavior that may indicate security breaches, fraud attempts, or other unusual activities at the company level.

Machine learning is a powerful tool for saving time spent on detecting suspicious activities within the network, increasing profitability and efficiency for certain departments or companies. In particular, it can detect insider threats, such as someone using stolen credentials to escalate their privileges, one of the most common types of attacks today. With the ability to recognize unusual behavior automatically, machine learning can detect user behavior changes, potentially indicating a stolen account or malicious activities performed by the user himself, helping organizations stay ahead of potential bad actors.

Aside from cybersecurity and user experience optimization, user behavioral analysis using machine learning is also critical in other domains, such as fraud detection. Machine learning algorithms can detect patterns of fraudulent behavior in financial transactions, insurance claims, or e-commerce activities, identifying suspicious behavioral patterns and preventing financial losses, mitigating risks, and protecting customers or users from fraudsters.

My personal experience with user behavior began during my internship last summer when I was assigned a project to develop, code, and integrate a PAM (Privilege Access Management) system to monitor actions performed by users with escalated privileges. While I was able to identify potential points of failure and good opportunities for research from a company perspective, machine learning was a new concept for me, leading to certain

complications. It took a long time to process the data, the data source was not fast nor useful enough, and I lacked experience with advanced algorithms for user behavior. Nonetheless, the algorithm I developed worked well for my level of software engineering expertise at the time.

In conclusion, the primary objective of this project is to monitor and analyze data. Data is crucial for understanding the main client group, identifying use cases for the project, and making data-driven decisions to enhance security measures, optimize user experience, and protect users, customers, and workers from potential risks or threats. With the computing power available today, it makes sense to invest time and effort in using machine learning to automate and improve efficiency, ensuring that the future of your project or company requires less time and effort spent on tasks that can be automated and done more precisely by a machine.

Dataset

My project is focused on analyzing user behavior, specifically in the realm of cybersecurity. Utilizing the BEHACOM dataset, which includes data from 12 users who used their computers without any limitations for 55 days. I was able to extract key features that can potentially indicate malicious activity. With over 50 features included in the dataset, I carefully selected the most relevant ones, including but not limited to **current_app, keystroke_counter, and changes_between_apps**, and others. By analyzing these features, my project aims to identify anomalies in user behavior that may indicate security breaches, fraud attempts, or other unusual activities at the company level.

My project will utilize the following:

- **current_app** - used to understand what kind of application is used in what way.
- **keystroke_counter** - how many symbols the user is inputting to understand patterns in typing.

- **erase_keys_counter** - if the user types exceptionally well, he will still use the backspace button for various reasons.
- **changes_between_apps** - how often switching between apps is performed.
- **press_press_average_interval** - the computer doesn't spend time pressing buttons, this can be a great indication
- **press_release_average_interval** - the computer will have them identify all over the place if it's not advanced enough.
- **received_bytes** - did it try to download something?
- **sent_bytes** - did it try to send something too large like the entire file system?
- **mouse_average_movement_speed** - people on average move the mouse the same way across all the applications.

However, an issue I faced was the lack of negative (not legit) results in the dataset. All inputs represent real user actions, meaning that none of them can be classified as bad. To solve this, I utilized the sample() method in '*pandas'* to generate fake information for testing and model evaluation. On average, I generated 300-1000 fake samples to enhance the validity of my findings. My personal experience with this project has taught me the importance of data-driven decision-making. By carefully selecting and analyzing the most relevant features, my project has the potential to provide valuable insights into user behavior, ultimately enhancing cybersecurity measures, optimizing user experience, and protecting users, customers, and workers from potential risks or threats.

Methods

Given the complexity of the problem, relying entirely on machine learning may not be sufficient. Therefore, in order to approach a more effective approach to classifying user actions

as either malicious or legit, I have designed a small, non-complex ranking system that can be revised for future use. This system involves assigning a weight, or multiplier, to each parameter discussed in the "Dataset" section based on its relative importance. By doing so, we can prioritize the most significant parameters in our analysis and ensure that our classification process is as accurate and efficient as possible.

List of factors:

1. **sent_bytes (factor = 5)**
2. **keystroke_counter (factor = 5)**
3. **erase_keys_counter (factor = 5)**
4. **recieved_bytes (factor = 4)**
5. **currrent_apps (factor = 3)**
6. **mouse_move_speed (factor = 2)**
7. **changes_between_apps (factor = 2)**
8. **press_release_average_interval (factor = 1)**
9. **press_press_average_interval (factor = 1)**

The reasoning behind choosing these specific parameters:

1. **current_app - understand what kind of application is used in what way**
2. **keystroke_counter - how many symbols the user is inputting to understand patterns is typing erase_keys_counter - if the user types exceptionally well, he will still use erase somehow, trust me**
3. **changes_between_apps - how often is trying to switch between apps**
4. **press_press_average_interval - how fast somebody physically presses buttons, the computer doesn't spend time on this**
5. **press_release_average_interval - computer will have them identify all over the place if it's not advanced enough**
6. **received_bytes - did it try to download something?**

7. **sent_bytes - did it try to send something too large like the entire file system?**
8. **mouse_average_movement_speed - people on the average move the mouse the same way across all the applications**

To ensure a more efficient and streamlined classification process, a ranking system was developed. Each parameter discussed in the "Dataset" section was assigned a factor, with 5 being the most important and 1 being the least. Using this system, each user's dataframe is input into the "analysis()" function which assigns a factor to each variable and generates a list for future reference. Each row of input is then analyzed, with the overall vulnerability ranking of each user action (row) calculated by summing the vulnerability potential of all elements within the row. The resulting ranking falls within the range of [0, 28], with 0 being the lowest ranking and 28 being the highest (most critical). To calculate the vulnerability ranking, the "vulnerability_ranking()" function was introduced. This function takes each value as an argument, calculates the quartiles, and determines how far from the mean the value is. Based on the quartile it falls into, the ranking for each element in the row is calculated. The resulting ranking is then used to determine the probability of vulnerability for each user action. The ranking is then calculated and passed back to determine whether the action vulnerability probability is:

- Legit
- Low
- Medium
- High
- Critical

Later, after the label is assigned, we would introduce **Linear Regression**. Linear regression is great due to the nature of that model, it's used to predict the value of a variable based on the other variables associated with it. This works great in our application.

Unfortunately, I wasn't able to tune it properly for higher accuracy, but the model works well based on my manual look-through. For output, we generate 2 files: both files are lists of events that are classified and differed as "*requires immediate action*" and "*blocked*" actions based on high and critical classification of actions respectively. After the data is processed and all desired actions are performed, we want to add a "**Classification**" column to the .csv file of original data for potential later analysis.

Conclusion

Given my accomplishments, while working on this project, I would continue working on it and implementing additional functionalities to enhance its stability, efficiency, and precision in classifying specific events. This project holds significant potential to solve some of the most pressing issues in the enterprise security sector, particularly with regard to user identification. In today's world, where scams and data breaches are on a rise, identifying users accurately and efficiently is more critical than ever before. By preventing compromised credentials, we can mitigate unpredictable damages and losses, both financially and in terms of reputation.

The machine learning model will need to be redone if not completely, but mostly. Due to not proper tune and the way it was setup, the machine learning part will be the most time-consuming in this project if I decide to go on with it.

Author links open overlay panelPedro M. Sánchez Sánchez a, a, b, c, and AbstractThis paper details the methodology and approach conducted to monitor the behaviour of twelve users interacting with their computers for fifty-five consecutive days without preestablished indications or restrictions. The generated dataset, "BEHACOM - A dataset modelling users' behaviour in computers," *Data in Brief*, 28-May-2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352340920306612.

[Accessed: 20-Apr-2023].